Modbus 转 MQTT Gateway

使用说明

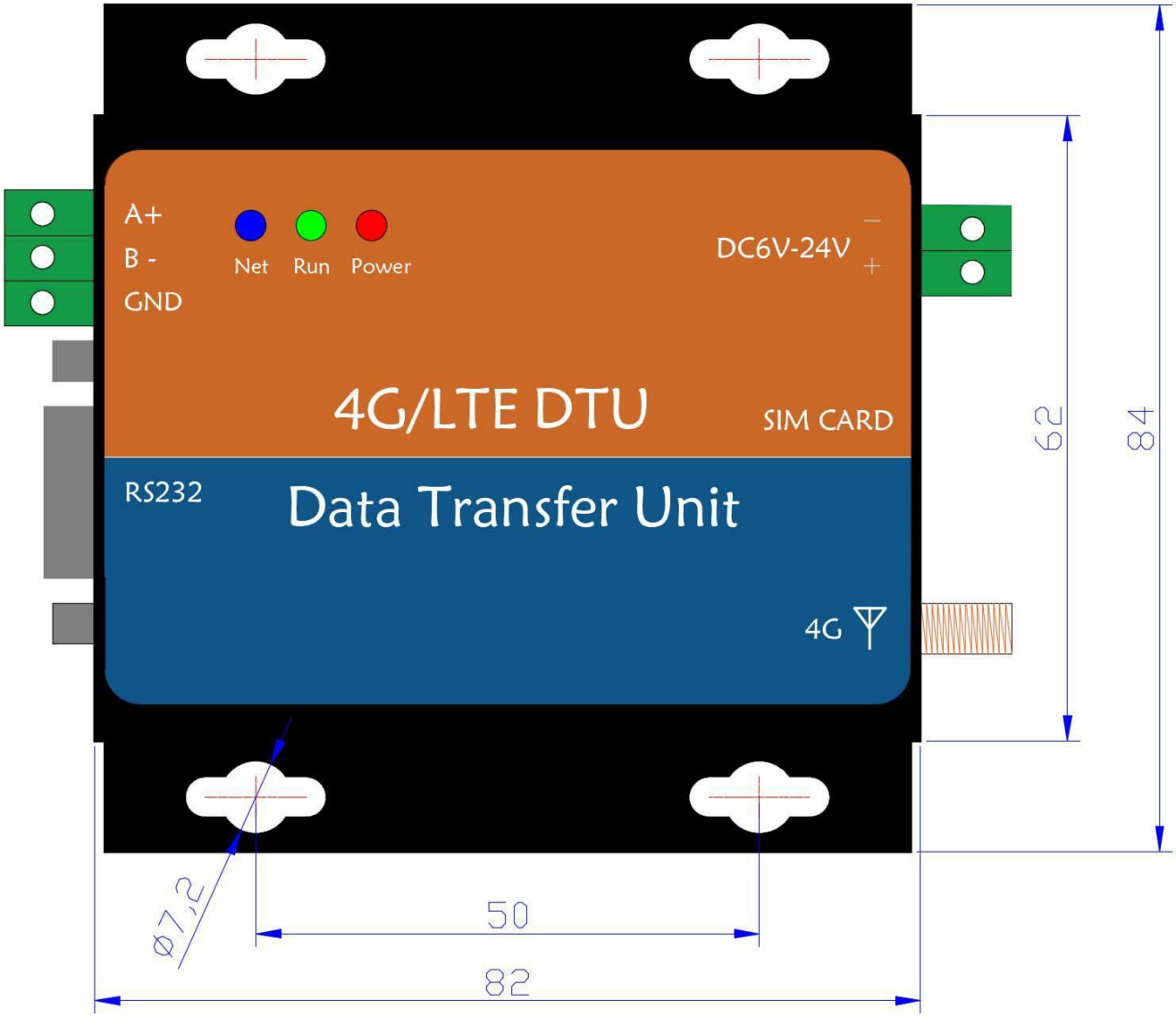
依云物联网

版本号：V3.1.2

目 录

一、网关模块的参数

1.1 模块的外观与安装尺寸（以 4G 模块为例，其它型号安装尺寸相同）：



1.2 GPRS 网关模块性能参数

GPRS 网关模块的性能参数		
项 目	**名 称**	**描述**
基本参数	储存温度	-40°C-85°C
	工作温度	-35°C-70°C
	工作电压	DC6V~24V

	工作电流	DC12V 空闲均值 40mA，发送数据时均值 55mA
硬件接口	GPRS 模组	移远 M26 工业级的四频全球通用 GSM/GPRS 无线模组
	MCU	ST 半导体，工业级 72MHz，32 位 ARM 内核处理器
	电源接口	2P-3.81 可插拔端子
	RS232 配置口	DB9 孔（2-RX,3-TX,5-GND），波特率 9600-115200bps
	RS485 数据口	A+,B- 3P-3.81 可拔插接线端子，波特率 1200-115200bps
	天线接口	SMA 阴头 *产品配套： 50Ω SMA 内针吸盘天线；
	SIM 卡槽	标准 3V/1.8V 推杆式 SIM 卡座，支持移动、联通大卡
网络参数	工作频段	GSM850,EGSM900,DCS1800,PCS1900
	发射功率	Class4(2W) GSM850,EGSM900； Class1(1W) DCS1800,PCS1900
	GPRS 数据特性	GPRS 数据下行传输：最大 85.6kbps； GPRS 数据上行传输：最大 85.6kbps
	编码格式	CS-1、CS-2、CS-3 和 CS-4

1.3 4G/LTE 网关模块性能参数

4G/LTE 网关模块的性能参数		
项 目	**名 称**	**描 述**
基本参数	储存温度	-40°C-85°C
	工作温度	-35°C-70°C
	工作电压	DC6V~24V
	工作电流	DC12V 空闲均值 52mA，发送数据时均值 95mA

硬件接口	LTE 模组	工业级 4G/LTE 无线模组(支持移动、联通、电信网络)
	MCU	工业级 72MHz，32 位 ARM 内核处理器
	电源接口	2P-3.81 可插拔端子
	RS232 配置口	DB9 孔（2-RX,3-TX,5-GND），波特率 9600-115200bps
	RS485 数据口	A+,B- 3P-3.81 可拔插接线端子，波特率 1200-115200bps
	天线接口	SMA 阴头 *产品配套： 50Ω 2dbi SMA 内针吸盘天线；
	SIM 卡槽	标准 3V/1.8V 推杆式 SIM 卡座，支持移动、联通、电信大卡
网络参数	工作频段	FDD-LTE: B1/ B3/； TDD-LTE: B38/ B39/ B40/ B41 WCDMA: B1 ； TD-SCDMA: B34/B39 EVDO/ CDMA2000 1x: BC0； GSM/EDGE: B3/8
	发射功率	Class 4 (33dBm±2dB) for GSM850 and EGSM900 Class 1 (30dBm±2dB) for DCS1800 and PCS1900 Class E2 (27dBm±3dB) for GSM850 and EGSM900 8-PSK Class E2 (26dBm±3dB) for DCS1800 and PCS1900 8-PSK Class 3 (24dBm+1/-3dB) for WCDMA and TD-SCDMA bands Class 3 (23dBm±2dB) for LTE FDD and TDD bands
	数据传输速率	FDD: Max 100Mbps (DL), Max 50Mbps (UL) TDD: Max 61Mbps (DL), Max 18Mbps (UL) 3GPP R6 HSUPA: Max 5.76Mbps (UL) 3GPP R8 DC-HSPA+: Max 42Mbps (DL) Support 3GPP R8 1.28 TDD Max 4.2Mbps (UL), 2.2Mbps

1.4 Ethernet 网关模块的性能参数

Ethernet 网关模块的性能参数

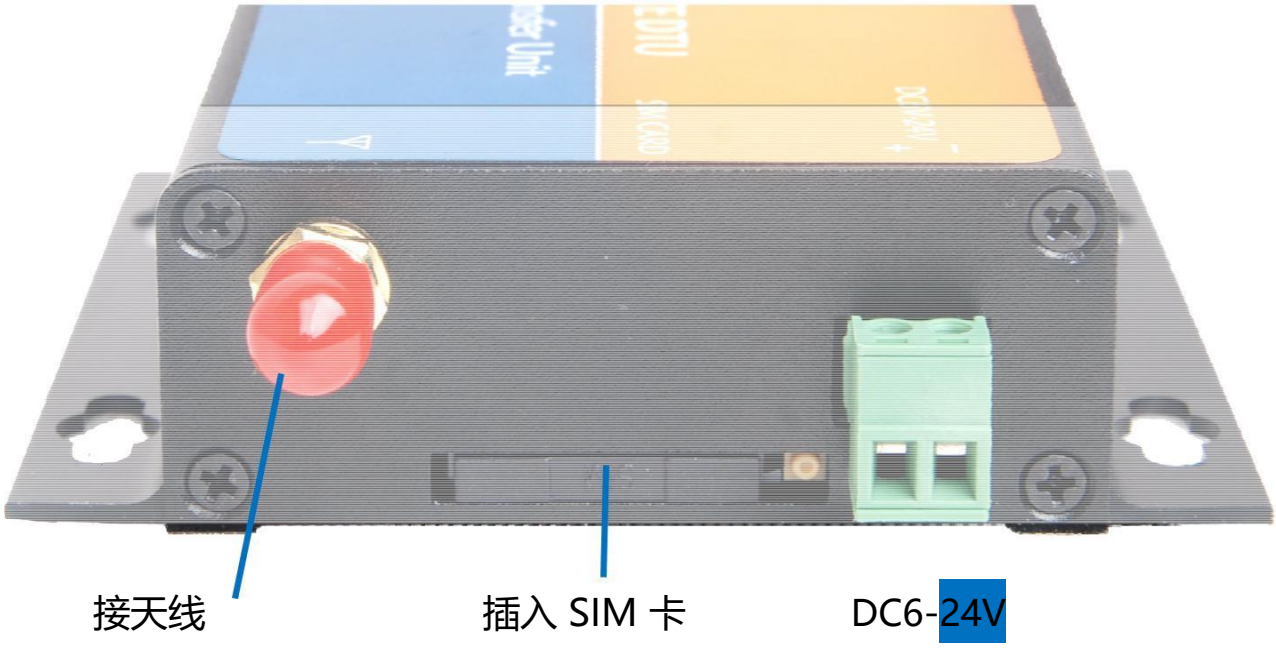
项 目	名 称	描 述
基本参数	储存温度	-40°C-85°C
	工作温度	-35°C-70°C
	工作电压	DC6V~24V
	工作电流	DC12V 均值 150mA
硬件参数	以太网接口	RJ45 10/100M
	MCU	工业级 72MHz，32 位 ARM 内核处理器
	电源接口	2P-3.81 可插拔端子
	RS232 配置口	DB9 孔（2-RX,3-TX,5-GND），波特率 9600-115200bps
	RS485 数据口	A+,B- 3P-3.81 可拔插接线端子，波特率 1200-115200bps

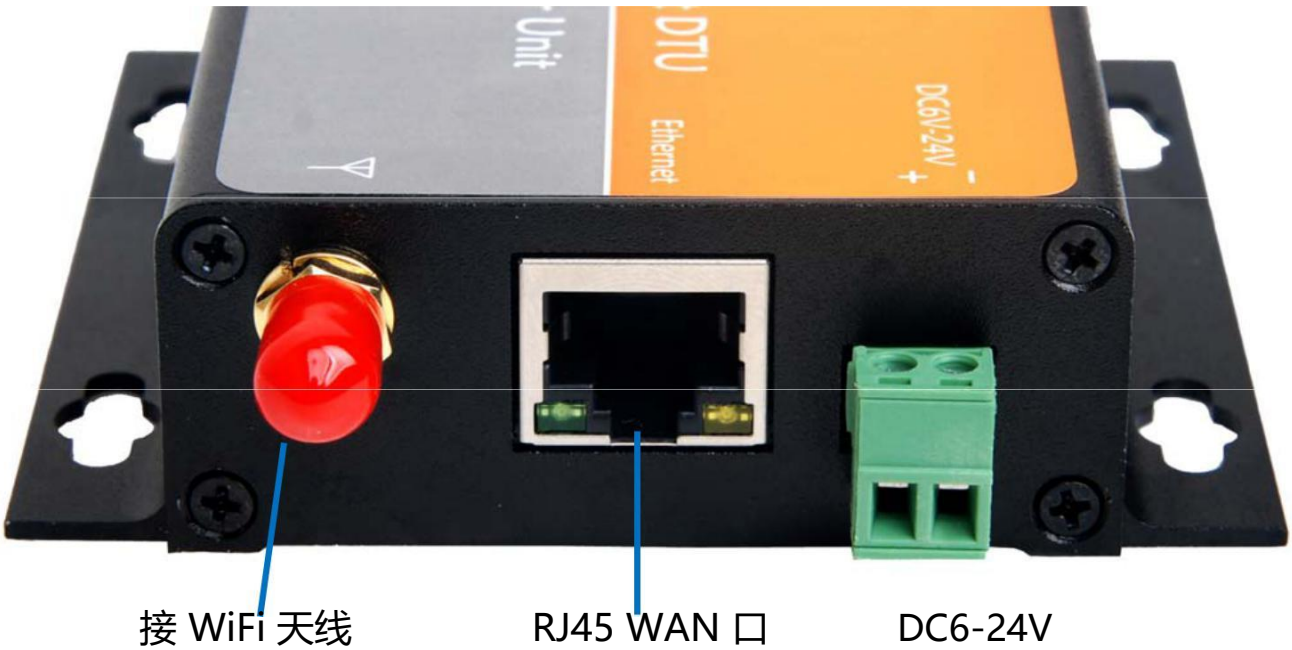
1.5 WiFi 网关模块的性能参数

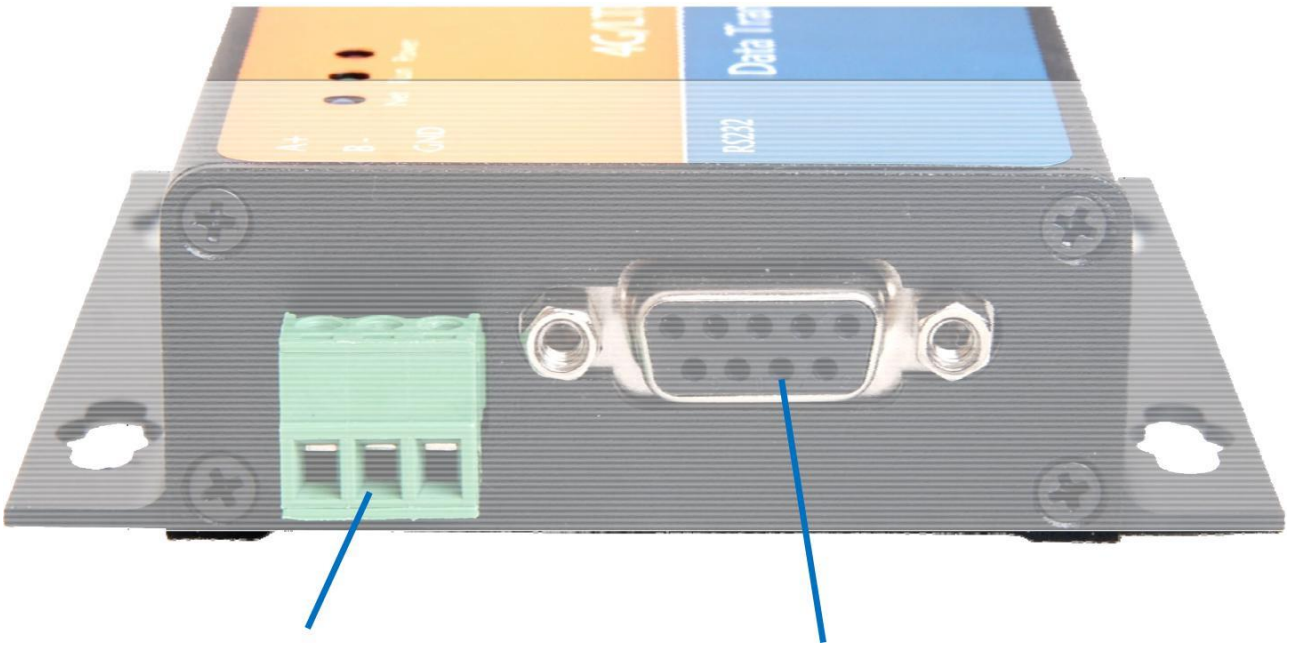
WiFi 网关模块的性能参数		
项 目	**名 称**	**描述**
基本参数	储存温度	-40°C-85°C
	工作温度	-35°C-70°C
	工作电压	DC6V~24V
	工作电流	DC12V 均值 150mA
硬件参数	WiFi 接口	802.11 b/g/n
	MCU	工业级 72MHz，32 位 ARM 内核处理器
	电源接口	2P-3.81 可插拔端子
	RS232	DB9 孔（2-RX,3-TX,5-GND），波特率 9600-115200bps
	RS485	A+,B- 3P-3.81 可拔插接线端子，波特率 1200-115200bps

1.4 模块的 LED 指示灯：

红色为 POWER（电源）灯，模块上电后亮起。蓝色灯为通讯模组状态灯，亮起或闪烁为正常。 绿色为 RUN（运行）灯，当模块注册到网络并连接到服务器后，RUN 灯开始交替闪烁，无交替闪烁时为断网状态。**二、 网关模块的连接**







RS485 端口（A+,B-,GND）　　配置口 RS232（2-RX,3-TX,5-GND）

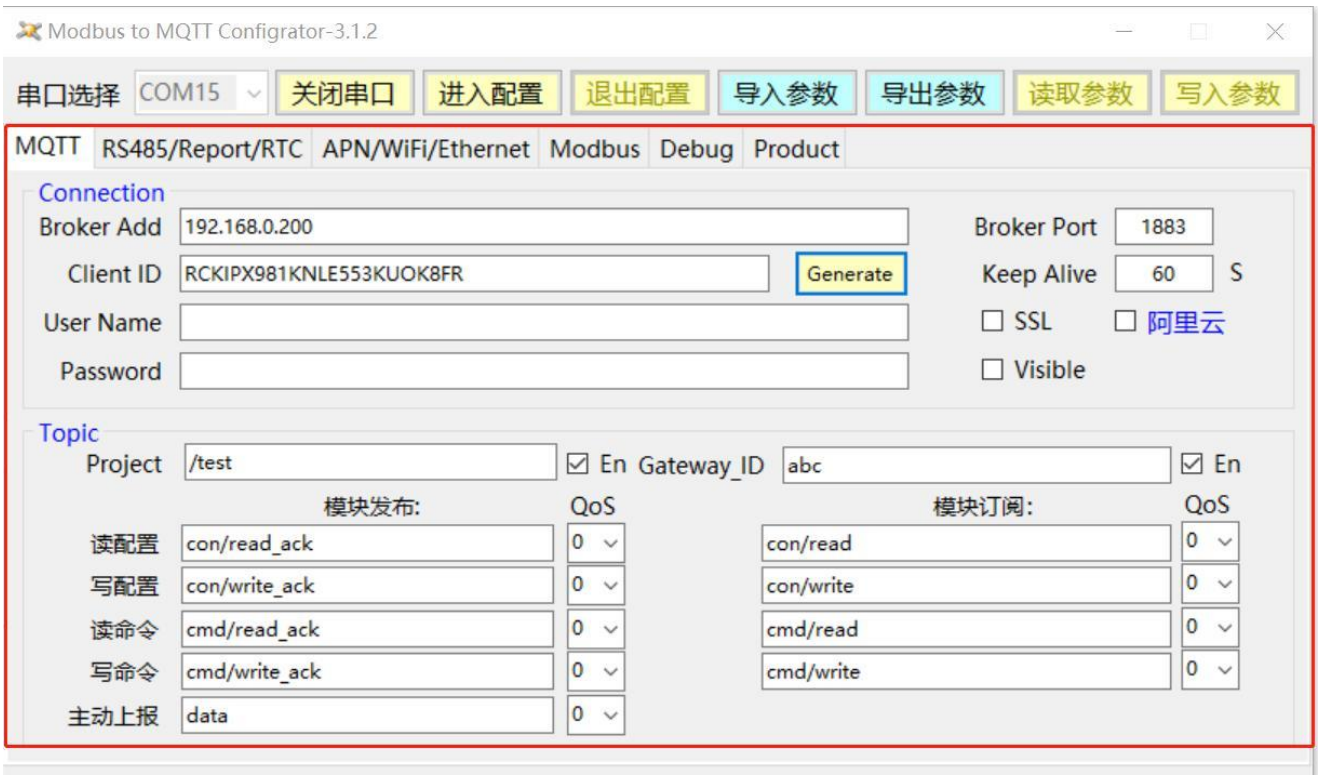
三、网关模块的参数配置

3.1，配置软件为绿色软件，在计算上打开该配置软件，如下图，串口未打开前，不能进行除“打开串口”外的其它任何操作；

3.2，在计算机上用 USB 转 RS232 DB9 公头数据通讯线直接连接到模块的 RS232 DB9 端口上后，打开软件，选择相应的串口号，然后点击“打开串口”，此时“进入配置”解锁亮起，点击“进入配置”按钮，等待软件自动搜索配置口通讯参数完成后，进入到配置模式，“进入配置”按钮被锁定，显示成灰色，“退出配置”按钮解锁亮起，如下图：

3.3 点击“读取参数”按钮，可以从模块中读出模块当前配置的参数，第一次打开时为出厂默认值。

3.4 MQTT 参数的配置， 打开 MQTT 选项卡，如下图：



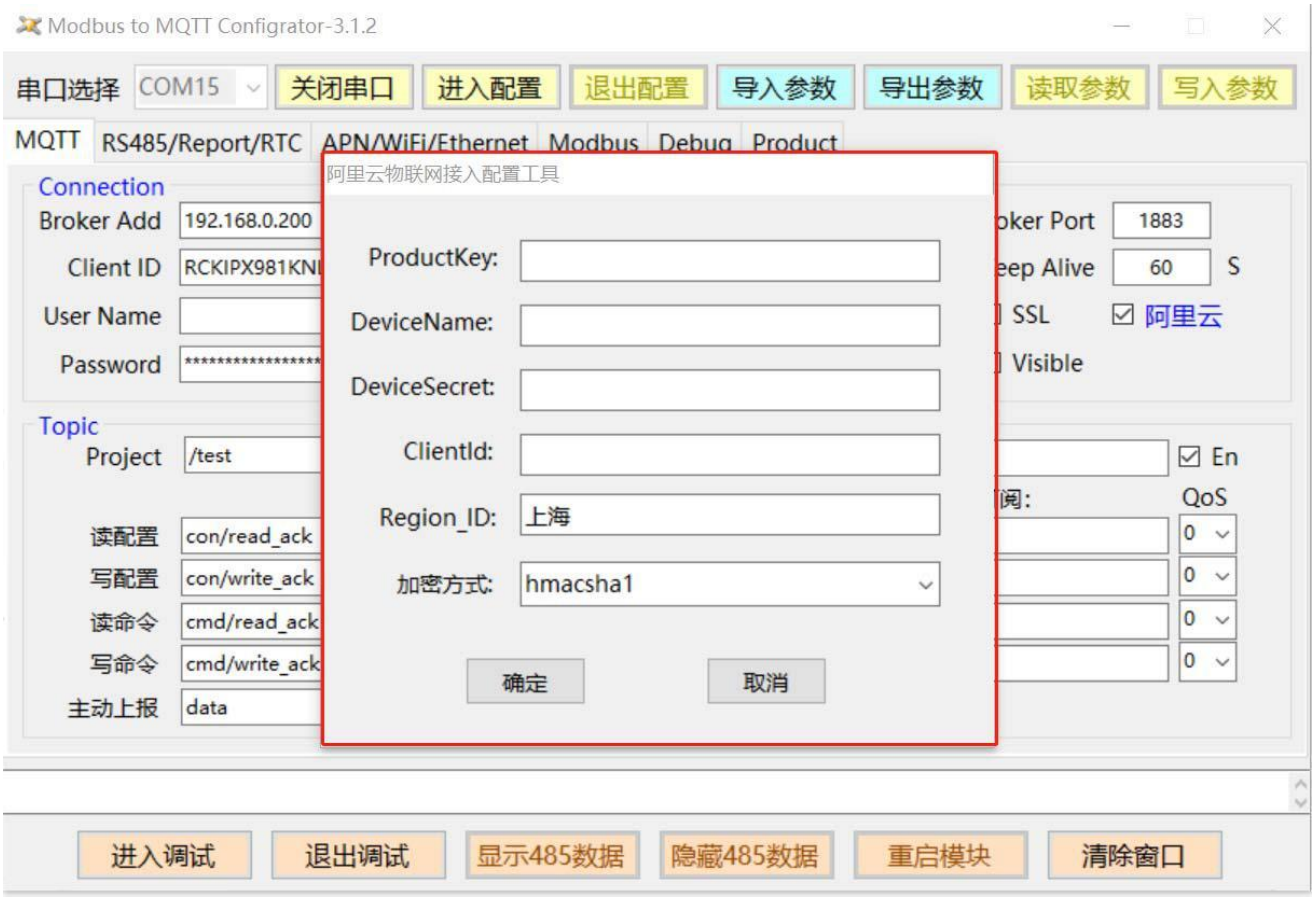
3.4.1 Connection

Broker Add 栏中填写运行 MQTT Broker 的 IP 地址或域名； Broker Port 栏中填写运行 MQTT Broker 的端口号；

Keep Alive 栏为保活时间间隔，以秒为单位，一般为设为 60S 即可; Clinet ID 栏中填写客户端 ID，即模块作为 MQTT 客户端的 ID，多个客户端（包含硬件和软件）ID 不能重复，可以点击“Generate”按钮生成随时 ID 号； User Name 栏中填写用户名（可以为空，取决于 MQTT Broker）；Password 栏中填写密码（可以为空，取决于 MQTT Broker）；

SSL 加密传输选项，目前仅支持 4G 全网通模块；

模块连接阿里云 Broker 时（如需要 SSL 时，必须先勾选 SSL，再勾选阿里云），可勾选“阿里云”复选框，弹出以下窗口：



根据阿里云提供的参数填入到相应的文本框，配置工具自动计算出阿里云的

Client ID 和 Password,并填入到相应文本框中。

3.4.2 Topic (主题)

根据 MQTT 协议对主题的说明，模块支持分层定义主题，具体如下：

Project 为模块发布订阅主题的最高层，不勾选 En 时，主题中不包含此文本框中的字符；

Gateway_ID 为模块发布订阅主题的第二层，不勾选 En 时，主题中不包含此文本框中的字符；

“读配置”为应用程序读取模块的配置信息时的主题。举例：应用程序读取模块的 485 参数时，如勾选了“Project”和“Gateway_ID”的 En，向模块订阅的主题为“Project”，“Gateway_ID”和读配置（模块发布列）对应文本框的字符串的拼接：“/test/abc/con/read”：（注：字符串可自定义）

```
{
"parameter":"485_par"
}
```

模块向主题“/test/abc/con/read_ack”返回数据：（注：字符串可自定义）

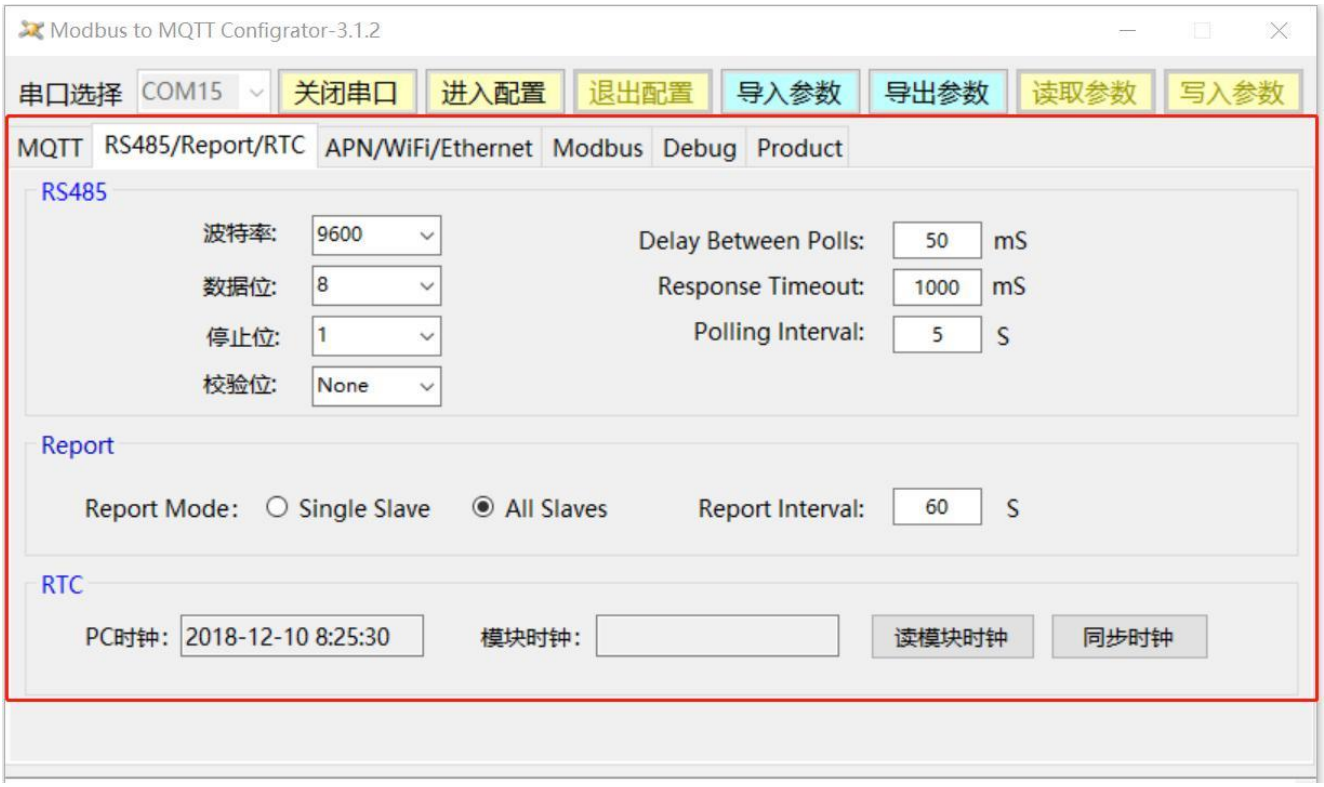
```
{
  "485_par" : {
    "polling_interval" : "2",
    "response_timeout" : "1000",
    "delay_between_poll" : "10",
    "485_uart" : [ "9600", "8", "None", "1" ]
  }
}
```

写配置、读命令、写命令与读配置类似，详见后续 Json 数据包说明；

主动上报主题为模块采集到的数据经解析后定时（可配置）上传时发布的主题，如：”/test/abc/data”，这个主题由“Project”，“Gateway_ID”和主动上报（模块发布列）的拼接而成，如不勾选“Project”和“Gateway_ID”的 En，则主动上报主题仅为主动上报（模块发布列）文本框中的字符，如：data；

Qos 为服务质量，网关模块支持 Qos0,Qos1,不支持 Qos2,根据需要选择；

3.5 RS485/Report/RTC 参数的配置， 打开此选项卡，如下图：



3.5.1 RS485 通讯参数：波特率 1200bps-115200bps，停止位 1-2，数据位 8，校验位有无，奇，偶校验等；

Delay Between Polls 为两条指令之间的间隔时间，建议 20mS 以上；

Response Timeout 为网关模块与从机通讯的超时时间，即当模块向从机发起轮询指令时，正常情况下，从机会立即返回数据，如果最长等待这个时间，从机仍未返回数据，模块将执行下一条轮询指令；

Polling Inverval 为模块主动发起轮询的时间间隔，即模块多长时间主动轮询一次的各个 Modbus 从机的数据，单位为秒；

3.5.2 Report 参数

Report Mode 为模块上报数据模式，” Single Slave”是以单个从机内的所有数据为一个数据包上报，一般有多个从机且数据量较大时采用此方式，” All Slave”是以网关接入的所有从机的所有数据为一个数据包上报，数据量不大时可采用此方式；

Report Inverval 为模块主动上报数据的时间间隔，即模块多长时间主动上报一次采集的 Modbus 从机的数据，单位为秒；

3.5.3 RTC 参数，对有实时时钟的模块，点击“读模块时钟”按钮可读出模块内的时钟，点击“同步时钟”按钮可以对模块进行对时；

3.6 APN/WiFi/Ehthernet 参数配置，如下图：

3.6.1 APN 参数（仅对 4G 全网通模块有效），如果 SIM 卡是公网卡，不需要设置，可以忽略此项，模块会自动根据中国移动、中国联通、中国电信等运营商配置相应的 APN 参数，如需要配置独立的 APN 参数，请填写此项，APN 栏填写运营商提供的“接入点名称”，User Name 栏和 Password 栏分别填写运营商提供的用户名和密码，UserName 和 Password 参数可以为空，根据运营商提供的信息填写。

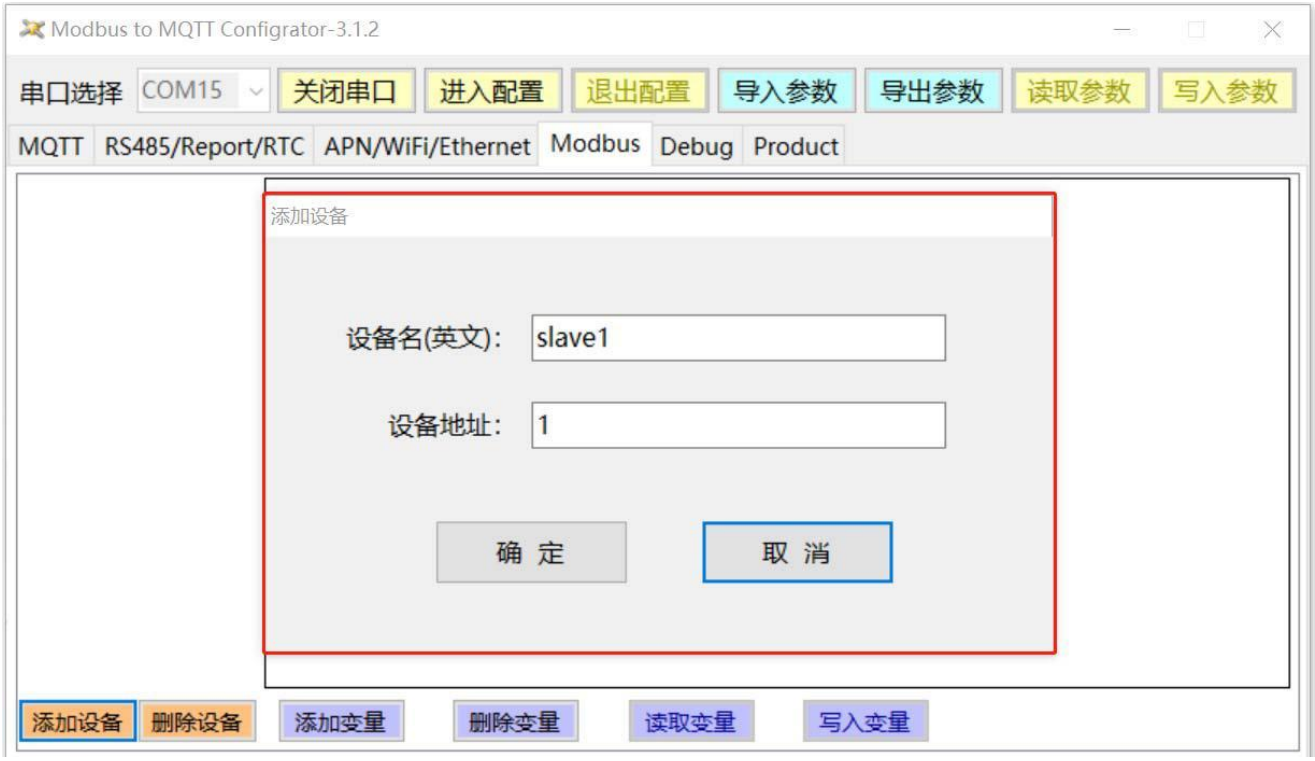
3.6.2 WiFi_SSID（仅对 WiFi 模块有效），配置模块要接入的 SSID 和密码;

3.6.3 Ethernet/WiFi IP 获取方式（对 Ethernet 和 WiFi 模块有效），模块具有 DHCP 功能，可自动获取 IP 地址，也可为模块配置固定的 IP 地址；

3.7 Modbus 参数配置，打开 Modbus 选项卡，如下图：

3.7.1 点击“读取参数”或“读取变量”按钮可从模块中读取当前的配置参数；

3.7.2 点击“添加设备”时弹出如下界面：



设备名为 Json 数据包中的设备键名（如："dev_name"："slave1"），设备

地址为 Modbus 从机的站号；

添加设备后显示界面如下，用户双击设备名称可修改其名称和站号：

3.7.3 删除设备：选中某个设备，点击“删除设备”即可删除该设备；

3.7.4 添加变量：选中某个设备后，然后点击“添加变量”按钮，弹出如下界面，进行变量添加（添加变量后，应用程序按 Json 格式读写即可，模块自动转化为 Modbus 指令，见 Modbus 协议和 Json 数据格式）：

“数据通道”包含 0 区-输出继电器（读写 bool 型，0 区的 Modbus 地址的首数字为 0，即 0xxxx，通道地址 0 对应的地址为 00001。以 01 功能码读取输出继电器的状态，以 05 功能码写继电器的状态，例如添加了从通道地址 0 开始，5 个继电器，从站地址为 1 时，模块自动生成“01 01 00 00 00 05 FC 09”16 进制指令码，定时轮询 1 号从机，从机返回数据后，模块对其解析并转换成 Json 格式数据后以 MQTT 协议发布到相应的主题上，应用程序订阅此主题后即可收到 5 个继电器的状态，应用程序以 Json 格式写入继电器 1 的状态时，模块将 Json 格式的数据转化为如“01 05 00 00 FF 00 8C 3A”的 16 进制指令发送到从机。），1 区-输入继电器（也称为输入状态，只读 bool 型，1 区的 Modbus 地址的首数字为 1，即 1xxxx，通道地址 0 对应的地址为 10001。以 02 功能码读取输入继电器的状态），3 区-输入寄存器（只读数值型，3 区的 Modbus 地址的首数字为 3，

即 3xxxx，通道地址 0 对应的地址为 30001。以 04 功能码读取输入寄存器的数值），4 区-保持寄存器（读写数值型，以 03 功能码读取保持寄存器的值，以 06 功能码写保持寄存器的值）。

“数据类型”支持 Bool（开关量型，值 0 与 1），Int16(有符号整型，值-32768 到 32767)，Uint16(无符号整型，值 0-65535)，Long ABCD（按 ABCD 顺序排列的 32 长整型），Long CDAB，Long BADC，Long DCBA，Float ABCD（按 ABCD 顺序排列的 32 浮点型），Float CDAB，Float BADC，Float DCBA 等数据类型；

“通道地址”是指要添加变量的地址或批量添加时的寄存器起始地址，0 区，1 区，3 区，4 区寄存器都是从 0 开始，如从保持寄存器 4 区的寄存器 0 开始读取一个整型数，对应的 Modbus 寄存器信息地址为 40001（也称 PLC 地址）；

“添加数量”是指一次需要添加变量的数量；

上图的含意为，一次添加 5 个保持寄存器，有符号数的变量；

左侧列表为添加的 5 个变量的信息，其中地址列显示要添加变量的寄存器信息地址，键名列显示对应寄存器上报数据时以 Json 格式的 key，采样低值（a），采样高值（b），工程低值（c），工程高值（d）为工程值转换参数，仅对 Int

和 Uint 类型有效，如采样值为 e，其转换公式为：转化值 f=(d-c)/(b-a)*(e-a)+c，具体如下：

40001 的键名修改为 tag1-1，工程值转换参数为默认值-32768-32767 对应

-32768-32767，转换值 f=(32767-(-32768))/ (32767-(-32768))*(e-0)+0=1e，即按采集的原值上报，不做数据处理；

40002 的变量名为 tag1-2，工程值转换为 0-10000 对应 0-100，转换值 f=(100-0)/(10000-0)*(e-0)+0=0.01e，即对采集的数据乘以 0.01 后上报，如采集到的值为 12345，上报时为 123.45；

40003 的变量名为 tag1-3，工程值转换为 0-4096 对应 0-16，根据公式，转换值 f=(16-0)/(4096-0)*(e-0)+0=0.0039e，如采集到的值为 1000，上报时为 3.91；

40004 的变量名为 tag1-4，工程值转换为 820-4096 对应 0-150，根据公式，f=(150-0)/(4096-820)*(e-820)-0，如采样值 e 为 1600，转换值为 35.7143，上报时精确到两位小数点，因此为 35.71；

40005 的变量名为 tag1-5，工程值转换为 820-32000（如 SIEMENS S7-200 PLC 的 AI 采样原值）对应-50-50，如采集到的值为 16000，根据工程值转换公式，f=(50-(-50))/(32000-6400)*(16000-6400)-50=-12.5；

变量参数配置完成后点击“确认“按钮，完成一种变量类型的添加，如有多种数据类型或寄存器地址不连续的变量，需多次添加。

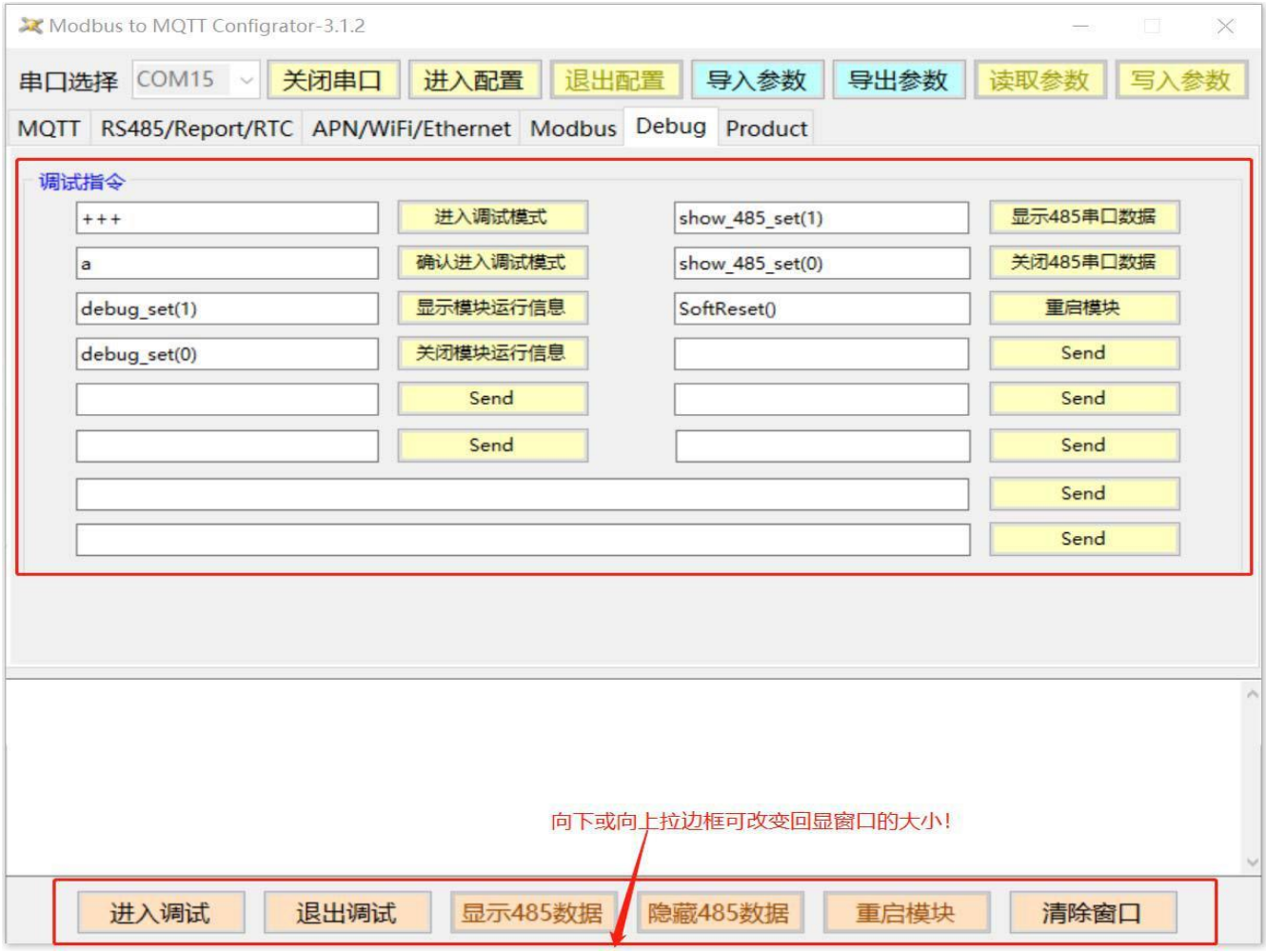
当添加的变量的数据类型为浮点型时，如下图，可以通过修改采样低值列中的数值改变上报浮点型数据的小数位，值为 1 时，即小数位精确到 1 位，如 123.43，上报时为 123.4；

3.7.5 删除变量：当选择一行或多行时已添加的变量后，点击“删除变量”删除当前选中的变量，如下图：

3.7.6 修改键名：添加好的变量，通过双击键名，采样值，工程值可对其进行修改操作；

3.7.7 当添加完变量后，点击“写入变量”或“写入参数”按钮，完成 Modbus 参数的配置。（注：仅写入 Modbus 参数时用“写入变量”，写入包含 Modubs 和其它全部参数时用“写入参数”）

3.8 模块调试，如下图：



打开串口后，在未进入配置的情况下，点击窗口底部的“进入调试“按钮，命令回显文本框中打印模块运行的信息；

在调试模式下，配置工具自动产生 Log 文件，在与配置工具同一文件夹下有一个与配置工具相同名称+年/月的文件夹，文件夹内按日期存储.TXT 日志；

“退出调试”后回显文本框中不打印模块运行的信息；

“显示 485 数据”在回显文本框中打印 485 的指令与返回数据；“隐藏 485 数据”在回显文本框中不打印 485 的指令与返回数据；

“重启模块”在调试模式下，点击此按钮可软重启模块；

“清除窗口”点击此按钮可清空回显文本框中的字符；

调试指令区为可以直接发指令给模块如：发送区：“+++”，“a”，“debug_set(1)”三条指令连续发送为进入到调试模式，点击对应的按钮可执行指令。当软重启模块或模块由于无法连接服务器自动软重启时，需用此指令组合再次进入到调试模式。其它指令的含义为：

“debug_set(1)”： 向串口打印调试信息；

“debug_set(0)”： 关闭向串口打印调试信息；

“show_485_set(1)”：向串口打印 Modbus 轮询指令及从机返回信息，点击“显示串口数据”按钮执行此指令，命令回显文本框中可显示向 Modbus 从机发送的轮询指令及从机返回的数据；

“show_485_set(0)”：不向命令回显文本框中打印 Modbus 轮询指令及从机返回信息；

3.9 固件升级

需要更新固件时，进入配置模式，然后打开“Product“选项卡，如下图：

点击“选择文件“按钮找到要升级的固件（固件类型为.bin），固件版本文本框中显示准备要升级的版本号；

”擦除配置“如擦除配置，待固件升级完成后需要重新配置模块参数，可以在更新固件前读出模块已配置的参数，等更新完参数后，将参数再写入到模块中；准备就绪后，点击”开始下载“按钮，进度条显示固件升级的进度，等下载完成后，模块自动重启生效，如需配置，再次点击”进入配置“按钮即可。

如果程序更新失败，无法再次进入到“配置模式”，请采用以下步骤更新：

1， 选择好需要更新的固件；

2， 给模块断电；

3， 给模块上电，并在 3S 内点击“开始下载”按钮；

4， 如进度条有变化，说明程序正在更新，等待更新完成即可，如点击了”开始下载”按钮没有成功，再次重复上述步骤。

四、网关模块的 Json 数据包格式说明

4.1 模块定时上报的数据消息：

模块定时上报数据时向“Project/Gateway_ID/data”**（这里的”Project”，“Gateway_ID”,”data”要替换成配置到模块相对应的参数）**主题发布从 Modbus 从机中采集到的数据，消息格式如下：

4.1.1 所有从机整体定时上报

```
{
"time" : "0000-00-00 00:00:00", //没有时钟的模块上报时全为 0;
"device" : [ {
"dev_name" : "slave1",
"comm_s" : "1",    //  与从机的通讯状态;
```

"variable" : {

"tag1-1":"31", //变量 tag1-1 及值：31，质量戳为 good

" tag1-2":"31", //变量 tag1-2 及值：31，质量戳为 good;

" tag1-3":"31", //变量 tag1-3 及值：31，质量戳为 good;

" tag1-4":"bad" //变量 tag1-4，无值，质量戳为 bad;

}

},{

"dev_name" : "slave2",

"comm_s" : "1",

"variable" : {

" tag2-1":"31", //变量 tag2-1 及值：31,，质量戳为 good

" tag2-2":"31", //变量 tag2-2 及值：31，质量戳为 good;

" tag2-3":"31", //变量 tag2-3 及值：31，质量戳为 good;

" tag2-4":"bad" //变量 tag2-4，无值，质量戳为 bad;

}

}]

}

4.1.2 单个从机定时上报

{

"time" : "0000-00-00 00:00:00", //没有时钟的模块上报时全为 0;

"dev_name" : "slave1",

"comm_s" : "1",

"variable" : {

```
"tag1-1":"31", //变量 tag11 及值：31,，质量戳为 good
" tag1-2":"31", //变量 tag12 及值：31，质量戳为 good;
" tag1-3":"31", //变量 tag13 及值：31，质量戳为 good;
" tag1-4":"bad" //变量 tag14，无值，质量戳为 bad;
    }
}
```

4.2 应用程序主动读取从机的数据

4.2.1 读取所有从机的数据时，应用程序向读命令主题如“Project： /Gateway_ID/cmd/read”**（这里的” Project”，“Gateway_ID”，” cmd/read”要替换成配置到模块相对应的参数）**主题发布以下消息：

```
{
  "dev_name":"all"，//读取数值所有从机;
  "var_name":"all" //读取数值所有变量;
}
```

网关向“Project/Gateway_ID/cmd/read_ack”**（这里的”Project”，“Gateway_ID”，” cmd/read_ack”要替换成配置到模块相对应的参数）**主题返回的所有从机的数据，消息格式参考第 4.1.1 节，模块定时上报的数据消息格式；

4.2.2 读取单个从机的数据时，应用程序向读命令主题如：“Project/Gateway_ID/cmd/read” **（这里的”Project”，“Gateway_ID”，” cmd/read” 要替换成配置到模块相对应的参数）**主题发布以下消息：

```
{
"dev_name":"slave1"，//读取指定从机的数据;
"var_name":"all" //读取指定从机的所有变量;
```

}

网关向“Project/Gateway_ID/cmd/read_ack” **（这里的”Project”，“Gateway_ID”，” cmd/read_ack”要替换成配置到模块相对应的参数）**主题返回数据，消息格式参考第 4.1.2 节，模块定时上报的数据消息格式；

4.2.3 读取指定从机的单个变量数据时，应用程序向读命令主题如：“Project/Gateway_ID/cmd/read” **（这里的”Project”，“Gateway_ID”，” cmd/read” 要替换成配置到模块相对应的参数）**主题发布以下消息：

```
{
"dev_name":"slave1", // 读取指定的从机;
"var_name":"tag1-1" // 读取指定的变量;
}
```

网关向“Project/Gateway_ID/cmd/read_ack” **（这里的”Project”，“Gateway_ID”，” cmd/read_ack”要替换成配置到模块相对应的参数）**主题返回数据，消息格式如下：

```
{
"dev_name" : "slave1",
"tag1-1" : "637.52"
}
```

4.3 向 Modbus 从机写入数值：

4.3.1 向 Modbus 从机写入值时，应用程序向“Project/Gateway_ID/cmd/write” **(这里的”Project”，“Gateway_ID”，” cmd/write”要替换成配置到模块相对应的参数）**主题发布以下消息：

```
{
```

```
"dev_name":"slave1",    // 对应的从机称为 slave1；
 "var_name":"tag1-1",        // 对应的变量名称为 T1；
 "value":"55"                // 对应的要写入变量的值为 55；
}
```

数据写入成功后，网关模块向“Project/Gateway_ID/cmd/write_ack” **（这里的”Project”，“Gateway_ID”，” cmd/write_ack”要替换成配置到模块相对应的参数）**主题发布以下消息：

```
{
"dev_name" : "slave1",
 "var_name:" : "tag1-1",
 " state" : "true"   //   如写入失败，返回 false
}
```

4.4 应用程序读取网关的 RS485 参数:

应用程序向“Project/Gateway_ID/con/read” **（这里的”Project”，“Gateway_ID”，” con/read”要替换成配置到模块相对应的参数）**主题发布以下消息：

```
{
"parameter":"485_par"
}
```

网关向 “Project/Gateway_ID/con/read_ack” **（这里的”Project”，“Gateway_ID”，” con/read_ack”要替换成配置到模块相对应的参数）**主题返回网关参数，消息格式如下：

```
{
  "485_par" : {
    "polling_interval" : "2",
    "response_timeout" : "1000",
    "delay_between_poll" : "10",
    "485_uart" : [ "9600", "8", "None", "1" ]
  }
}
```

4.5 应用程序写入网关的 RS485 参数:

应用程序向“Project/Gateway_ID/con/write” **（这里的”Project”，“Gateway_ID”，”con/write”要替换成配置到模块相对应的参数）**主题发布以下消息：

```
{
  "485_par" : {
    "polling_interval" : "2",
    "response_timeout" : "1000",
    "delay_between_poll" : "20",
    "485_uart" : [ "9600", "8", "None", "1" ]
  }
}
```

网关向“Project/Gateway_ID/con/write_ack” **（这里的”Project”，“Gateway_ID”，”con/write_ack”要替换成配置到模块相对应的参数）**主题返回网关参数，消息格式如下：

```
{
  "485_par" : {
    "polling_interval" : "2",
    "response_timeout" : "1000",
    "delay_between_poll" : "20",
    "485_uart" : [ "9600", "8", "None", "1" ],
    "state" : "success"
  }
}
```

4.6 应用程序读取网关的主动上报参数:

应用程序向“Project/Gateway_ID/con/read” **（这里的”Project”，“Gateway_ID”，”con/read”要替换成配置到模块相对应的参数）**主题发布以下消息：

```
{
"parameter":"report_par"
}
```

网关向 “Project/Gateway_ID/con/read_ack” **（这里的”Project”，“Gateway_ID”，”con/read_ack”要替换成配置到模块相对应的参数）**主题返回网关参数，消息格式如下：

```
{
  "report_par" : {
    "report_interval" : "30",
    "report_mode" : "all"
  }
}
```

4.7 应用程序写入网关的主动上报参数:

应用程序向“Project/Gateway_ID/con/write” **（这里的”Project”，“Gateway_ID”，”con/write”要替换成配置到模块相对应的参数）**主题发布以下消息：

```
{
  "report_par" : {
    "report_interval" : "50",
    "report_mode" : "all"
  }
}
```

网关向“Project/Gateway_ID/con/write_ack” **（这里的”Project”，“Gateway_ID”，”con/write_ack”要替换成配置到模块相对应的参数）**主题返回网关参数，消息格式如下：

```
{
  "report_par" : {
    "report_interval" : "50",
    "report_mode" : "all",
    "state" : "success"
  }
}
```

4.8 应用程序读取网关的 Modbus 参数:

应用程序向“Project/Gateway_ID/con/read” **（这里的”Project”，“Gateway_ID”，”con/read”要替换成配置到模块相对应的参数）**主题发布以下消息：

```
{
"parameter":"modbus_par"
 }
```

网关向 “Project/Gateway_ID/con/read_ack” **（这里的”Project”，“Gateway_ID”，”con/read_ack”要替换成配置到模块相对应的参数）**主题返回网关参数，消息格式如下：

第一包数据：

```
{
  "modbus_par" : {
    "var_num" : "18"    //共 18 个变量;
  }
}
```

第二包数据（5 个变量）：

```
 {
"modbus_par" : {
```

```
        "data_start" : "0",
        "data_end" : "4",
        "data" : {
           "0" : [ "slave1", "1", "     1", "DO1", "0", "0.0", "0.0", "0.0", "0.0" ],
           "1" : [ "slave1", "1", "     2", "DO2", "0", "0.0", "0.0", "0.0", "0.0" ],
           "2" : [ "slave1", "1", "     3", "DO3", "0", "0.0", "0.0", "0.0", "0.0" ],
           "3" : [ "slave1", "1", "     4", "DO4", "0", "0.0", "0.0", "0.0", "0.0" ],
           "4" : [ "slave1", "1", "10001", "DI1", "0", "0.0", "0.0", "0.0", "0.0" ]
        }
     }
  }
```


第三包数据（5 个变量）

第四包数据（5 个变量）

第五包数据（3 个变量）

注：如果变量数超过 5 个，模块将变量分为 5 个一包进行上报，最后一包不足 5 个的以实际为准；

4.9 应用程序写入网关的 Modbus 参数:

应用程序向“Project/Gateway_ID/con/write” **（这里的”Project”，“Gateway_ID”，”con/write”要替换成配置到模块相对应的参数）**主题发布以下消息：

发第一包数据（写入变量总数量）：


```
{
   "modbus_par" : {
      "var_num" : "18"   //共 18 个变量;
   }
}
```


模块向“Project/Gateway_ID/con/write_ack” **（这里的”Project”，“Gateway_ID”，”con/write_ack”要替换成配置到模块相对应的参数）**主题返回：


```
{
   "modbus_par" : {
      "varnum" : "18",
      "state" : "success"
   }
}
```


发第二包数据（写入前 5 个变量）：

```
{
  "modbus_par" : {
    "data_start" : "0",
    "data_end" : "4",
    "data" : {
      "0" : [ "slave1", "1", " 00001", "DO1", "0", "0.0", "0.0", "0.0", "0.0" ],
      "1" : [ "slave1", "1", " 00002", "DO2", "0", "0.0", "0.0", "0.0", "0.0" ],
      "2" : [ "slave1", "1", " 00003", "DO3", "0", "0.0", "0.0", "0.0", "0.0" ],
      "3" : [ "slave1", "1", " 00004", "DO4", "0", "0.0", "0.0", "0.0", "0.0" ],
      "4" : [ "slave1", "1", "10001", "DI1", "0", "0.0", "0.0", "0.0", "0.0" ]
    }
  }
}
```


模块向“Project/Gateway_ID/con/write_ack” **（这里的”Project”，“Gateway_ID”，”con/write_ack”要替换成配置到模块相对应的参数）**主题返回：


```
{
  "modbus_par" : {
    "data_start" : "0",
    "data_end" : "4",
    "state" : "success"
  }
}
```


发第三包数据（下 5 个变量）：


```
{
  "modbus_par" : {
    "data_start" : "5",
    "data_end" : "9",
    "data" : {
      "5" : [ "slave1", "1", "10002", "DI2", "0", "0.0", "0.0", "0.0", "0.0" ],
      "6" : [ "slave1", "1", "10003", "DI3", "0", "0.0", "0.0", "0.0", "0.0" ],
      "7" : [ "slave1", "1", "10004", "DI4", "0", "0.0", "0.0", "0.0", "0.0" ],
      "8" : [ "slave1", "1", "10005", "DI5", "0", "0.0", "0.0", "0.0", "0.0" ],
      "9" : [ "slave1", "1", "10006", "DI6", "0", "0.0", "0.0", "0.0", "0.0" ]
    }
  }
}
```


模块向“Project/Gateway_ID/con/write_ack” **（这里的”Project”，“Gateway_ID”，”con/write_ack”要替换成配置到模块相对应的参数）**主题返回：


```
{
  "modbus_par" : {
    "data_start" : "5",
    "data_end" : "9",
```

```
      "state" : "success"
    }
  }
```

第四包数据（下 5 个变量）：

```
{
  "modbus_par" : {
    "data_start" : "10",
    "data_end" : "14",
    "data" : {
      "10" : [ "slave1", "1", "30001", "AI1", "1", "820", "4096", "0", "100" ],
      "11" : [ "slave1", "1", "30002", "AI2", "1", "820", "4096", "0", "100" ],
      "12" : [ "slave1", "1", "30003", "AI3", "1", "820", "4096", "0", "100" ],
      "13" : [ "slave1", "1", "30004", "AI4", "1", "820", "4096", "-0", "100" ],
      "14" : [ "slave1", "1", "40001", "AO1", "1", "0", "4096.0", "0.0", "100" ]
    }
  }
}
```

模块向“Project/Gateway_ID/con/write_ack” **（这里的”Project”，“Gateway_ID”，”con/write_ack”要替换成配置到模块相对应的参数）**主题返回：

```
{
  "modbus_par" : {
    "data_start" : "10",
    "data_end" : "14",
    "state" : "success"
  }
}
```

第五包数据（最后 3 个变量）：

```
{
  "modbus_par" : {
    "data_start" : "15",
    "data_end" : "17",
    "data" : {
      "15" : [ "slave2", "2", "40001", "AI2-1", "1", "820", "4096", "0", "100" ],
      "16" : [ "slave2", "2", "40002", "AI2-2", "1", "820", "4096", "0", "100" ],
      "17" : [ "slave2", "2", "40003", "AI2-3", "1", "820", "4096", "0", "100" ]
    }
  }
}
```

模块向“Project/Gateway_ID/con/write_ack” **（这里的”Project”，“Gateway_ID”，”**

con/write_ack”要替换成配置到模块相对应的参数）主题返回：

```
{
  "modbus_par" : {
    "data_start" : "15",
    "data_end" : "17",
    "state" : "success"
  }
}
```

第六包数据（保存变量并生效）：

```
{
  "modbus_par" : {
    "data_over" : "true"
  }
}
```

模块向“Project/Gateway_ID/con/write_ack” **（这里的”Project”，“Gateway_ID”，”**

con/write_ack”要替换成配置到模块相对应的参数）主题返回：

```
{
  "modbus_par" : {
    "data_over" : "true",
    "state" : "success"
  }
}
```

五、网关模块的 Modbus RTU 协议说明

5.1 模块支持的 MODBUS RTU 协议部分功能码：

功能码	功能码意义	可操作的寄存器寻址地址	可操作的寄存器信息地址	位/字操作	模块是否支持
01	读继电器状态	0x0000-0x270F	00001-09999	位操作	支持
02	读(开关)输入状态	0x0000-0x270F	10001-19999	位操作	支持
03	读保持寄存器	0x0000-0x270F	40001-49999	字操作	支持
04	读输入寄存器	0x0000-0x270F	30001-39999	字操作	支持
05	写单个继电器	0x0000-0x270F	00001-09999	位操作	支持
06	写单个保持寄存器	0x0000-0x270F	40001-49999	字操作	支持

16	写多个保持寄存器	0x0000-0x270F	40001-49999	字操作	支持

5.2，Modbus 寄存器的信息地址与寻址地址之前的关系：

寄存器类型	寄存器信息地址	寄存器寻址地址	读写状态	功能码
0 区-输出继电器	00001-09999	0x0000-0x270F	可读可写	01，05
1 区-输入继电器	10001-19999	0x0000-0x270F	只读	02
3 区-输入寄存器	30001-39999	0x0000-0x270F	只读	04
4 区-保持寄存器	40001-49999	0x0000-0x270F	可读可写	03，06

5.2.1 存器信息地址（PLC 地址）

寄存器信息地址指的是存放变量状态或数据的 Modbus 从机寄存器的地址，Modbus 从机可以是 PLC，RTU，触摸屏，数显表、数采仪，传感器等。例如 00005、10009、40001、30003 等，这些地址一般使用十进制描述，可以分别描述不同寄存器类型的地址。

5.2.2 寄存器寻址地址（协议地址）

寄存器寻址地址指的是通信时使用的寄存器地址，例如信息地址 40001 对应寻址地址 0x0000，40002 对应寻址地址 0x0001，寄存器寻址地址一般使用 16 进制描述。再如，信息寄存器 40003 对应寻址地址 0X0002，信息寄存器 30003 对应寻址地址 0X0002，虽然

两个信息寄存器通信时使用相同的地址，但是需要使用不同的功能码（03，04）才可以访问，所以访问时不存在冲突。

5.3，寄存器类型说明：

寄存器类型	说明	举例
输出继电器	输出端口，按位操作，可设定端口的输出状态，也可以读取该输出状态。	继电器，电磁阀，晶体管，LED 灯等的输出状态。
输入继电器（输入状态）	输入端口，按位操作，通过外部改变输入状态，可读不可写。	拨码开关，接近开关，机械开关等的输入状态。

输入寄存器	输入端口，控制器运行时从外部设备获得的数据。	-5V,0-10V,4-20mA ,PT100 等模拟量信号的输入。
保持寄存器	内部数据或输出端口数据，可读可写。也常见将传感器输入的数据存放到保持寄存器中，做为只读数据。	模拟量输出设定值，运行参数，AD 转换采样参数等数据。也常见外部传感器。

5.4，举例说明：

由配置软件添加了一个 1 号从站，8 个继电器（寄存器信息地址 00001-00008），5 个保持寄存器（寄存器信息地址 40005-40009）,模块运行时自动生成两条查询指令：

1, 读输出继电器　HEX(01 01 00 00 00 08 3D CC)　// 00001 对应的寻址地址为 00 00

	HEX	备注
从机地址	01	
功能码	01	由于功能码为 01，因此起始地址（寻址地址）对应寄存器的信息地址为 00001
寄存器起始地址高位	00	
寄存器起始地址低位	00	
寄存器数量高位	00	
寄存器数量低位	08	
CRC 校验高位	3D	
CRC 校验低位	CC	

2, 读保持寄存器　HEX(01 03 00 04 00 05 3D CC)　// 40005 对应的寻址地址为 00 04

当应用程序要打开 00001 继电器时，模块生成指令：

HEX (01 05 00 00 00 FF 8C 3A)　　// 00001 对应的寻址地址为 00 00

当应用程序向 40006 写入 11 时，模块生成指令：

HEX (01 06 00 05 00 0B D8 0C)　　// 00001 对应的寻址地址为 00 05