

1.BC26 硬件板准备.....	2
2.UDP 测试介绍.....	3
3.TCP 测试介绍.....	5
4.MQTT 测试介绍.....	6

墨子号科技

1.BC26 硬件板准备

申明:

我们提供回环测试服务器, 保证用户可以在没有服务器的情况下, 测试模块是否正常。

UDP: IP:47.92.146.210 PORT:9999

TCP: IP:47.92.146.210 PORT:8888

MQTT: IP:47.92.146.210 PORT:1883

(1) 拿到 BC26 模块板之后, 用户需要接入 USB TTL 模块对板子进行调试, 注意板子上有 PWR 接 3V3 的跳线帽, 请不要拔掉, 主要是为了让模块开机能够自启动。正常上电, 板子上的指示灯会亮。表明上电正常。

USBTTL 接线说明, USBTTL 上的跳线帽也不要动:

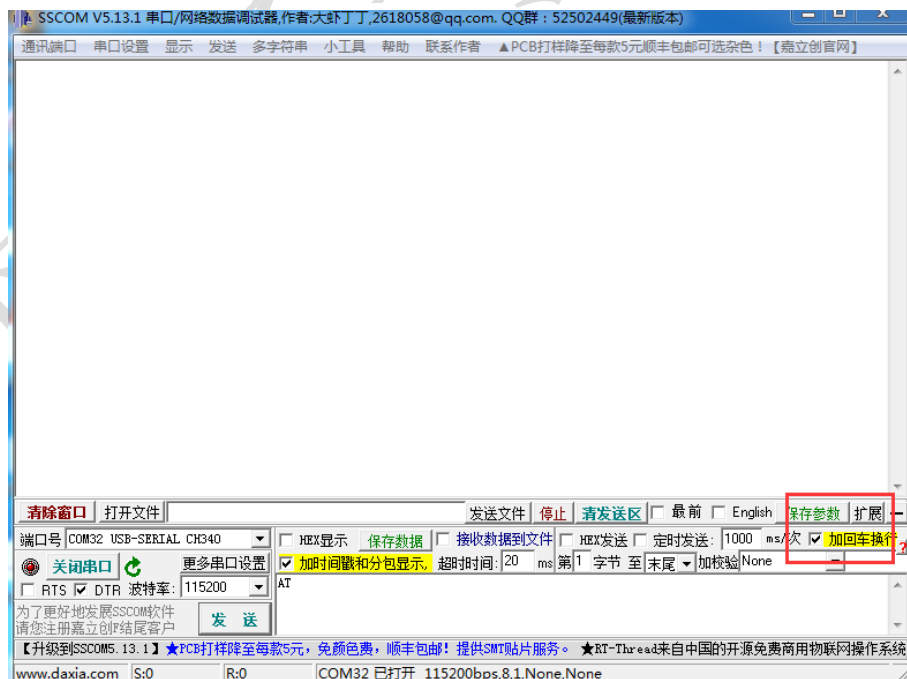
5V-VIN

GND-GND

RXD-TXD

TXD-RXD

(2) 打开串口调试助手或者 QCOM 软件, 波特率请务必选择 115200。然后用户可以发送 AT 指令去控制, 注意回车换行请务必也要勾选。



(3) 那用户可以先发几个指令测试下模块是否正常。首先可以先发 AT, 如果发 AT 发现返回 OK, 表明模块当前是好的, 是可以正常实现收发的。如果出现不返回数据, 那么就要查看是不是 PWR 没有接好, 串口波特率以及回车换行等是否处理得当。

AT 返回 OK,代表模块正常。

```
[15:17:27.917]发→◇AT
□
[15:17:27.928]收←◆
OK
```

AT+CIMI 是查询是否有卡，如果有卡，会返回 460，如果没有卡，会返回 ERROR。

```
[15:18:01.557]发→◇AT+CIMI
□
[15:18:01.567]收←◆
460042760503248
OK
```

AT+CSQ 是查询信号的，如果注册网络成功，会有信号产生，最大 31，如果信号小于 10，说明当前网络信号不佳。

```
[15:18:23.758]发→◇AT+CSQ
□
[15:18:23.766]收←◆
+CSQ: 31,0
OK
```

AT+CGATT?是查询注册网络情况，如果注册成功，会返回 1，如果失败，返回 0。

```
[15:21:20.478]发→◇AT+CGATT?
□
[15:21:20.489]收←◆
+CGATT: 1
OK
```

上面几条指令，只要正常了。那么用户就可以实现 UDP, TCP, LWM2M, COAP 等相关的协议开发了。当前比如其他的指令获取 IP 等此时都是可以实现了。用户在使用的时候，上面的指令一定要满足。如果有不满足的情况，那么下面的正式开放就没法正式使用了。

2.UDP 测试介绍

UDP 测试用户可以参考 BC26 的” [Quectel_BC26_TCPIP_AT_Commands_Manual_V1.0_20180711](#)” 或者更新的资料。里面有相关的介绍。

前面的一切就绪之后，就可以去连接 UDP 服务器。当然此处需要有服务器支持才可以。如果没有服务器用户可以先用我们的来进行使用。接入 IP：

47. 92. 146. 210 端口：9999。发什么就会回复什么数据。可以确保测试正常。

```
AT+QIOPEN=1,0,"TCP","220.180.239.212",8062,0,1 //Context 1 and <connectID> is 0.
OK
+QIOPEN: 0,0 //Connect successfully. It is suggested to wait 60 seconds for the
connection result.
```

用户参考这个指令来测试，只要将图中的 TCP 改成 UDP 即可。然后后面跟的分别是 IP 和端口（服务器）就可以测试了。

```
[16:54:48.647]发→◇AT+QIOPEN=1,0,"UDP","47.92.146.210",9999,0,1
□
[16:54:48.657]收←◆AT+QIOPEN=1,0,"UDP","47.92.146.210",9999,0,1
OK
[16:54:48.695]收←◆
+QIOPEN: 0,0
```

上面就是发对接服务器的指令，如果输入指令没有错误，会返回 OK 的。返回的 QIOPEN:0,0，表明连接上了，不过需要注意的是 UDP 是无连接模式，哪怕当前的端口不存在，他依然会显示连接成功。所以这一点用户需要注意。TCP 会检测。UDP 并不会。

```
[16:31:22.289]发→◇AT+QISEND=0,10,1234567890
□
[16:31:22.331]收←◆
OK
[16:31:22.358]收←◆
SEND OK
[16:31:23.075]收←◆
+QIURC: "recv",0,10
1234567890
```

上面是发送指令函数，同时 AT 指令里面手册也有，他支持字符串和十六进制 2 种方式发送数据。

因为我们的服务器支持自动下发。所以返回的数据有 URC 上报，可以看到数据发送成功并且接收到服务器下发的数据。表明 UDP 收发测试完成。

3.TCP 测试介绍

TCP 测试用户可以参考 BC26

的” [Quectel_BC26_TCPIP_AT_Commands_Manual_V1.0_20180711](#)” 或者更新的资料。里面有相关的介绍。

前面的一切就绪之后，就可以去连接 TCP 服务器。当然此处需要有服务器支持才可以。如果没有服务器用户可以先用我们的来进行使用。接入 IP：

47. 92. 146. 210 端口：9999。发什么就会回复什么数据。可以确保测试正常。

```
AT+QIOPEN=1,0,"TCP","220.180.239.212",8062,0,1 //Context 1 and <connectID> is 0.
OK

+QIOPEN: 0,0 //Connect successfully. It is suggested to wait 60 seconds for the
connection result.
```

用户参考这个指令来测试，只然后后面跟的分别是 IP 和端口（服务器）就可以测试了。

```
[16:55:34.118]发→◇AT+QIOPEN=1,0,"TCP","47.92.146.210",8888,0,1
□
[16:55:34.128]收←◆AT+QIOPEN=1,0,"TCP","47.92.146.210",8888,0,1
OK
[16:55:34.922]收←◆
+QIOPEN: 0,0
```

上面就是发对接服务器的指令，如果输入指令没有错误，会返回 OK 的。返回的 QIOPEN:0,0，表明连接上了。返回的数据一定要是 0,0 后面的数据是 0，如果不是 0，那么多数是接入有问题。要查服务器，因为 TCP 是三次握手。他连接的 TCP 服务器必须是真实有效的。

```
[16:31:22.289]发→◇AT+QISEND=0,10,1234567890
□
[16:31:22.331]收←◆
OK
[16:31:22.358]收←◆
SEND OK
[16:31:23.075]收←◆
+QIURC: "recv",0,10
1234567890
```

上面是发送指令函数，同时 AT 指令里面手册也有，他支持字符串和十六进制 2 种方式发送数据。

因为我们的服务器支持自动下发。所以返回的数据有 URC 上报，可以看到数据发送成功并且接收到服务器下发的数据。表明 TCP 收发测试完成。

4.MQTT 测试介绍

MQTT 测试用户可以参考 BC26 的”[Quectel_BC26_MQTT_Application_Note_V1.0](#)”或者更新的资料。里面有相关的介绍。

其实 MQTT 现在都是内置在了模块里面，对于用户而言相对简单了很多，不需要再去关心底层使用情况。只要实现登录到服务器即可。

MQTT 其实是 TCP 上面做了一些协议封装，本质与 TCP 没有区别。用户可以去研究 MQTT 底层实现功能，我们这里做不做过多的展开，主要是给用户来介绍下如果来接入 MQTT 服务器实现数据的收发。

不过需要注意的是 MQTT 里面的收发有专业术语。

发是发布（publish），收是订阅（subscribe）。

同样 MQTT 也是需要服务器的，不过这个服务器一般不需要用户去搭建，可以选择用我们的，因为服务器主要做个数据转发功能，他不参与数据的介入。换句话说服务器不会对我们发数据，只会对发来的数据进行检测，从而返回相关的处理流程给到设备这边。最大的特色就是支持了模块板与模块板之间的数据收发功能。

下面介绍下接入流程。

BC26 的手册也给出了 example，用户可以参考例程一步步的进行接入。

6 Examples

This chapter gives an example to explain how to use MQTT related commands.

```

AT+QMTOPEN=?
+QMTOPEN: (0-5), "<host_name>", <port>

OK

//Open a network for MQTT client.
AT+QMTOPEN=0, "220.180.239.212", 8401
OK

+QMTOPEN: 0,0           //Opened the MQTT client network successfully.

AT+QMTOPEN?
+QMTOPEN: 0, "220.180.239.212", 8401

OK

AT+QMTCONN=?
+QMTCONN: (0-5), "<clientId>", "<username>", "<password>[]"

OK

AT+QMTCONN=0, "clientExample"
OK

+QMTCONN: 0,0,0         //Connected the client to MQTT server successfully.

AT+QMTSUB=?
+QMTSUB: (0-5), <msgID>, "<topic>", <qos>[, "<topic>", <qos>...]

```

首先需要先连接上服务器，这个是第一步，因为不接入服务器，那一切也是枉然。所以我们先将设备接入到服务器，然后再继续工作。

那么这里我们提供 MQTT 服务器给用户免费测试，用户可以接入我们提供的服务器。IP:47.92.146.210, PORT:1883。

```

[16:56:31.726]发→◇AT+QMTOPEN=0, "47.92.146.210", 1883
□
[16:56:31.741]收←◆AT+QMTOPEN=0, "47.92.146.210", 1883
[16:56:31.766]收←◆
OK
[16:56:32.439]收←◆
+QMTOPEN: 0,0

```

好，我们输入登录服务器的指令之后，会显示 QMTOPEN:0,0, 这个与 TCP 是一个概念。返回的最后一个参数一定要是 0，如果是其他数据，表明接入的服务

器有错误，或者操作有问题，需要检查。这个接入成功之后，就可以真正的登录服务器了。

MQTT 其实与普通的 TCP 就多一个区别，因为 MQTT 为了方便管理用户，就加了一个登录约束条件。就是说每一个用户登录服务器的时候，需要发送一个 ID 给到服务器，这个是基本要求。有的服务器是需要账号和密码的。比如阿里云就是的，要求就很多。我们当前提供的服务器，只需要有一个 ID 即可。账号密码无所谓，有没有都能正常的使用。

注意这个 ID 是唯一的。不能与其他的设备重合，因为 MQTT 服务器对设备的管理主要是通过 ID 来进行管理。如果说 ID 重合，那么就会出现混乱的情况。所以这一点大家也需要注意。

所以一般推荐用户连接服务器发送 ID 的时候，这个 ID 可以选用模块的 IMEI 号作为登录 ID，这样就可以保证不会重复了。

```
[12:19:23.410]发→◇AT+QMTCONN=0,"clientExample"
[12:19:23.425]收←◆AT+QMTCONN=0,"clientExample"
[12:19:23.477]收←◆
OK
[12:19:23.670]收←◆
+QMTCONN: 0,0,0
```

这个指令就是发送 ID 到服务器进行审核，如果说审核通过了，就会返回 0, 0, 0，如果返回的不是这三个 0，表明审核失败。那用户就要查问题了，看下是不是需要账号密码，还是连接断开等问题了。

好，登录成功之后，用户就可以随意的进行数据发布与订阅了。这个就是类似 TCP 的数据发送与接收一个概念。

那么首先做一下数据发布，看下数据发布的情况。

```
[12:23:57.027]发→◇AT+QMTPUB=0,0,0,0,"mzh_bc26","hello MQTT."
[12:23:57.036]收←◆AT+QMTPUB=0,0,0,0,"mzh_bc26","hello MQTT."
[12:23:57.065]收←◆
OK
+QMTPUB: 0,0,0\0
```

上面就是使用 MQTT 协议进行数据发布，这个就完成了整个 MQTT 从登录到连接到发布的整个流程完成。其实很简单，就三句话。发布数据需要注意的是

“mzh_bc26”他是发布主题，什么意思呢，就是你发送数据出去，你得告诉别人你的数据是发给谁的。就是你发快递给别人得告诉快递公司这个货物是谁来签收的。那么这个主题的概念就是一样的。快递里面的包裹东西就是你所发布的数据内容了。

“hello MQTT.”这个就是发布的数据内容了。

所以发布数据 2 要素，一个是主题收件人，一个是包裹内容，这个与发快递是一个概念。大家这样理解就非常容易了。

那么这个数据发送到哪里去了呢。这里介绍 2 个方法，比如我们可以自发自收。因为 MQTT 我们知道他是支持收发的，所以呢，我们也可以自己订阅自己发布的数据，MQTT 支持用户这样操作。所以我们订阅下 “mzh_bc26” 这个主题看下，是否自己可以收到数据。

```
[12:31:33.955]发→◇AT+QMTSUB=0,1,"mzh_bc26",0
[12:31:33.964]收←◆AT+QMTSUB=0,1,"mzh_bc26",0
[12:31:33.990]收←◆
OK
[12:31:34.690]收←◆
+QMTSUB: 0,1,0,0
```

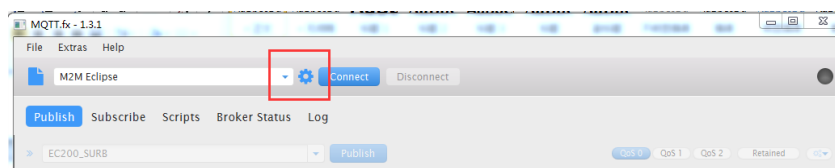
好，我们看到已经订阅了这个主题，并且显示订阅成功，因为后面的数字是 0，但是会疑惑怎么没数据呢，因为你前面发布的数据我没有订阅，那肯定是没法显示的，只有你再次发送发布数据的时候，才会有显示，好的测试下：

```
[12:33:12.444]发→◇AT+QMTPUB=0,0,0,0,"mzh_bc26","hello MQTT."
[12:33:12.453]收←◆AT+QMTPUB=0,0,0,0,"mzh_bc26","hello MQTT."
[12:33:12.481]收←◆
OK
+QMTPUB: 0,0,0\0
[12:33:13.223]收←◆
+QMTRECV: 0,0,"mzh_bc26","hello MQTT."
```

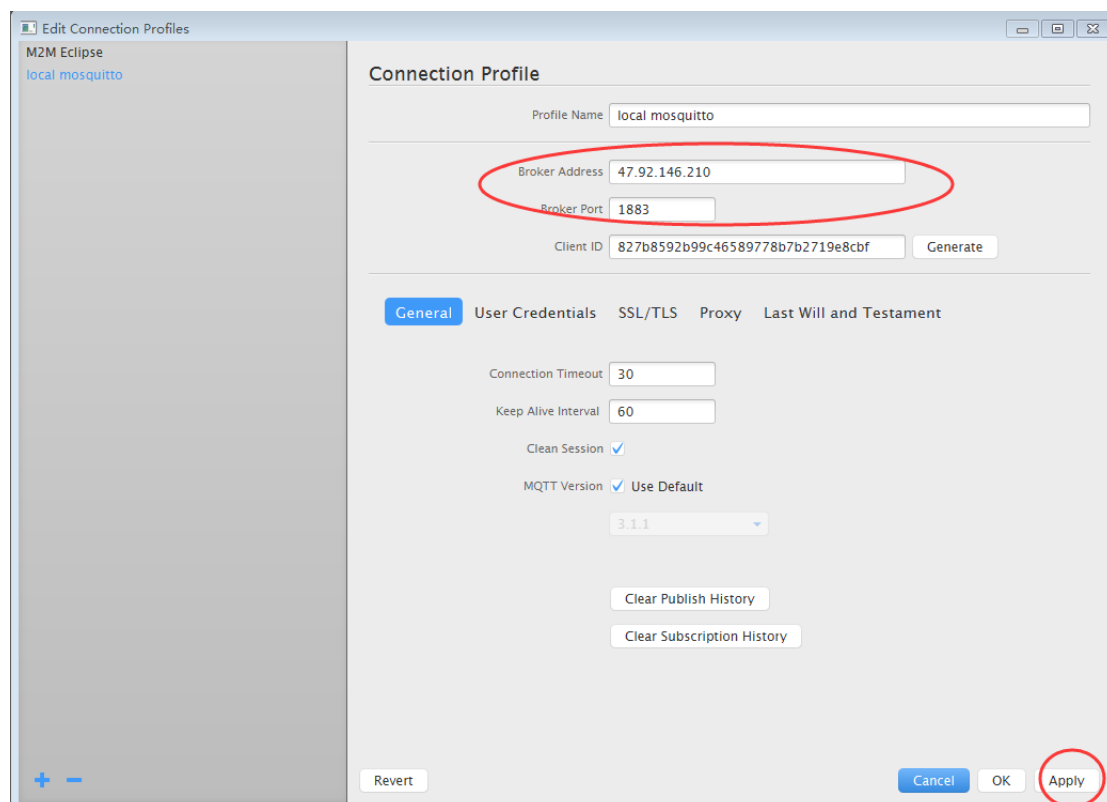
看下此时发布数据的同时，下面有 URC 上报，将订阅的主题和内容进行了上报。这个就表明当前的 MQTT 测试收发是正常的了。

此外还有个方法就是通过 mqtt 的软件来实现在自己的电脑或者手机端进行数据的订阅与发布。这里推荐用户使用 mqtt.fx 这个软件，最新应该有到 1.7 的版本了。

用户在自己的电脑上面装一个 mqtt.fx 软件，然后打开启动使用即可。

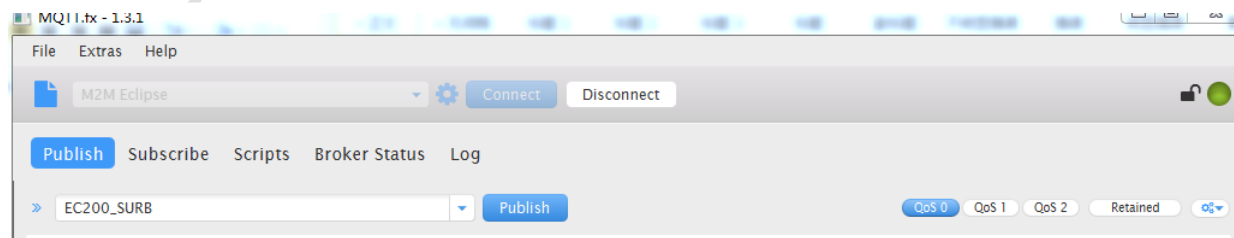


这个地方需要配置下，进入把服务器的 IP 地址改一下。



注意 IP 地址一定要改成我们所提供的这个，他自带的 IP 也可以用，只是有的时候会出现连接失败的问题，所以建议用户将这个服务器 IP 改成我们的，并 Apply 一下。

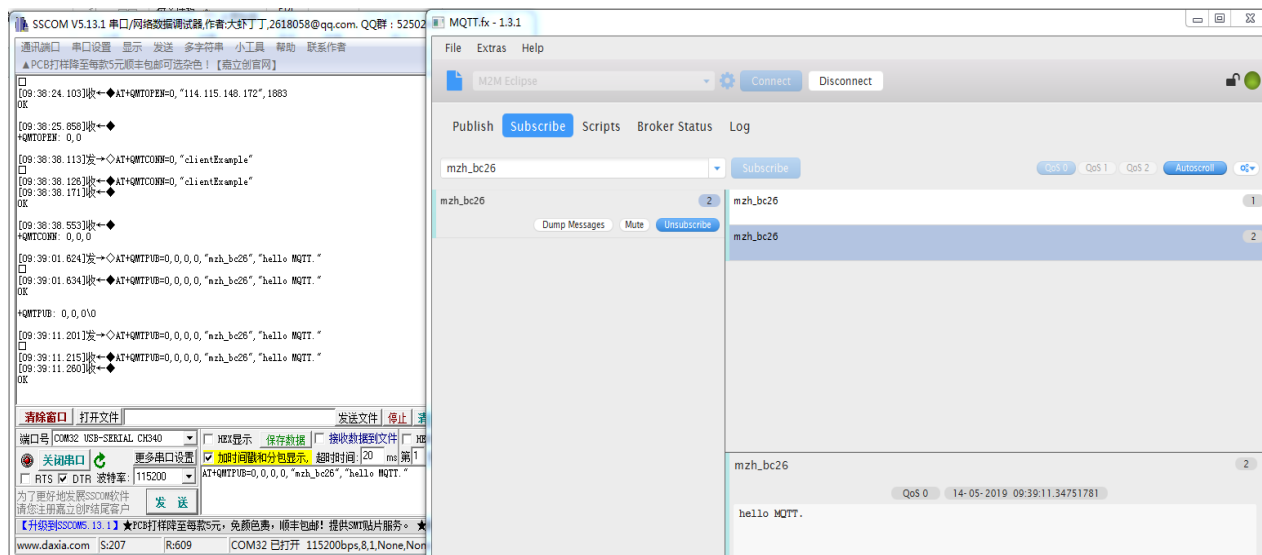
然后就可以进行连接测试。



点击链接，就可以了。如果显示绿色，表明接入是成功的。

下面有个选项，一个是发布的英文一个是订阅的英文。用户此时就可以进行发布和订阅数据了。

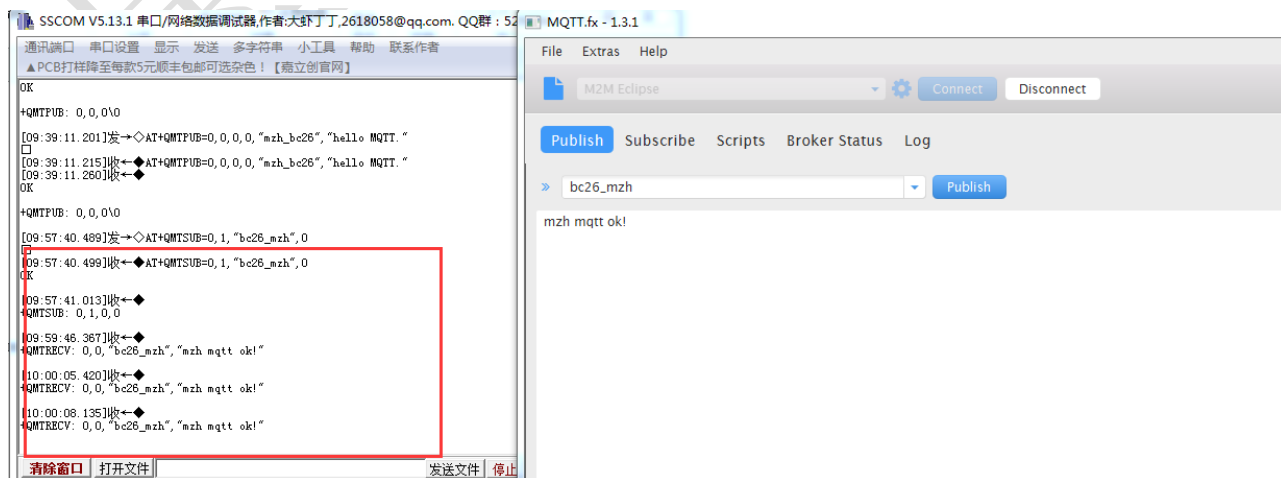
我们首先订阅个主题。上面的模块发布的主题是” mzh_bc26”，那么这里我们也要订阅这个主题，换做其他主题是没法订阅的，这要注意。



上面这张图就可以清晰的看到，左边是串口调试发送指令进行 MQTT 数据发布，针对的是模块。右边是 mqtt.fx 软件订阅到数据所显示的画面。从右边可以看到订阅的主题与发布的主题一样，同时数据内容与模块发布的数据消息也是一样。这样，软件就实现了订阅模块数据发布的功能了。

同时那么下面就可以进行模块的订阅，软件的数据发布了。

我用模块订阅一个” bc26_mzh” 主题，然后看下 mqtt.fx 发布这个主题和数据看下设备的数据接收情况。



从上面的图可以得到，设备订阅了主题，并显示订阅成功，当 mqtt.fx 下发数据的时候，因为订阅与发布主题一样，所以就收到了数据，并进行了显示。从而验证了模块与软件的收发正常。

