

WiFi 模块

# BMC81M001

## Arduino Library V1.0.3 说明

版本: V1.20 日期: 2023-12-20

[www.bestmodulescorp.com](http://www.bestmodulescorp.com)

## 目录

简介 .....	3
<b>Arduino Lib 函数</b> .....	<b>3</b>
<b>Arduino Lib 下载及安装</b> .....	<b>8</b>
<b>Arduino 范例</b> .....	<b>9</b>
范例 1: TCP .....	9
范例 2: Alinyun_Iot .....	11
范例 3: ThingSpeak .....	17
范例 4: ThingSpeakPublish .....	23
范例 5: ThingSpeakSubscribe .....	29

## 简介

BMC81M001 是倍创推出的 WiFi 模块，使用 UART 通信方式。本文档对 BMC81M001 的 Arduino Lib 函数、Arduino Lib 安装方式进行说明；范例演示了 TCP 和阿里云平台以及 ThingSpeak 的数据传输。

## Arduino Lib 函数

Arduino Lib 名称: BMC81M001		Lib 版本: V1.0.3
<b>构造函数 &amp; 初始化</b>		
1	<b>BMC81M001(HardwareSerial*theSerial = &amp;Serial)</b>	
	描述	构造函数，选择硬件 UART 通信方式
	参数	* theSerial: 硬件 UART 通信接口选择
	返回值	—
	备注	—
2	<b>BMC81M001(uint16_t rxPin,uint16_t txPin)</b>	
	描述	构造函数，选择软件 UART 通信方式
	参数	rxPin: 软件 UART 的 RX 引脚 txPin: 软件 UART 的 TX 引脚
	返回值	—
	备注	—
3	<b>void begin(uint32_t baud = BMC81M001_baudRate)</b>	
	描述	模块初始化
	参数	baud: 波特率，默认 115200
	返回值	void
	备注	—
<b>功能函数 (TCP)</b>		
4	<b>bool connectToAP(String ssid,String pass)</b>	
	描述	模块连接 WiFi
	参数	ssid: WiFi 名称 pass: WiFi 密码
	返回值	执行情况: true: 成功 false: 失败
	备注	—
5	<b>bool connectTCP (String ip,int port)</b>	
	描述	接入 TCP 服务器
	参数	ip: TCP 服务端的 IP 地址 port: 端口号
	返回值	执行情况: true: 成功 false: 失败
	备注	IP 地址和端口号由服务器定义

6	<b>bool writeDataTcp(int Dlength, char Dbuffer[])</b>	
	描述	向 TCP 服务器发送数据
	参数	Dlength: 数据长度 Dbuffer[]: 数据
	返回值	执行情况: true: 成功 false: 失败
	备注	—
7	<b>String readDataTcp ()</b>	
	描述	连接 TCP 服务器, 读取服务器发送的数据
	参数	—
	返回值	TCP 服务器发送的数据
	备注	—
<b>功能函数 (Iot)</b>		
8	<b>bool configMqtt(String clientlid,String username,String password,String mqtt_host,int server_port);</b>	
	描述	配置 MQTT 连接参数
	参数	clientlid: 客户端、用户 ID。 username: 用户名称。 password: 密码。 mqtt_host: 服务器地址。 server_port: 服务器端口。
	返回值	执行情况: true: 成功 false: 失败
	备注	—
9	<b>bool setPublishTopic(String publishtopic);</b>	
	描述	设置发布的默认 Topic
	参数	IoT 平台上 Topic 列表中的默认 publish Topic
	返回值	执行情况: true: 成功 false: 失败
	备注	—
10	<b>bool setSubscribetopic(String subscribetopic);</b>	
	描述	设置订阅的默认 Topic
	参数	IoT 平台上 Topic 列表中的默认 subscribe Topic
	返回值	执行情况: true: 成功 false: 失败
	备注	—

11	<b>bool setTopic(String topic)</b>	
	描述	设置 IoT 平台的自定义 topic
	参数	<b>topic:</b> 消息发布者和订阅者之间的传输中介
	返回值	执行情况: <b>true:</b> 成功 <b>false:</b> 失败
	备注	—
12	<b>bool writeString (String Dbuffer,String topic)</b>	
	描述	采用 String 数据格式向 IoT 平台发送数据
	参数	<b>Dbuffer:</b> 要发送的字符型数据 <b>topic:</b> 消息发布者和订阅者之间的传输中介
	返回值	执行情况: <b>true:</b> 成功 <b>false:</b> 失败
	备注	—
13	<b>bool writeBytes (char Dbuffer[],int Dlength,String topic)</b>	
	描述	采用字节数据格式向 IoT 平台发送数据
	参数	<b>Dbuffer[]:</b> 数据 <b>Dlength:</b> 数据长度 <b>topic:</b> 消息发布者和订阅者之间的传输中介
	返回值	执行情况: <b>true:</b> 成功 <b>false:</b> 失败
	备注	—
14	<b>void readIotData (String *RecvBuff,int *RecvBufflen,String *topic)</b>	
	描述	连接 IoT 平台后，读取平台发送的数据
	参数	<b>*RecvBuff:</b> 接收到的数据 <b>*RecvBufflen:</b> 接收数据长度 <b>*topic:</b> 接收到的数据所属 Topic
	返回值	void
	备注	—
<b>其他函数</b>		
15	<b>bool reset(void)</b>	
	描述	软件复位模块
	参数	void
	返回值	执行情况: <b>true:</b> 成功 <b>false:</b> 失败
	备注	—

16	<code>int sendATCommand(String StringstrCmd,int timeout,uint8_t reTry);</code>	
	描述	发送 AT 指令
	参数	StringstrCmd: AT 指令 timeout: 超时时间 reTry: 超时次数
	返回值	执行情况: true: 成功 false: 失败
	备注	—

注 1: 登录阿里云物联网平台, 进入控制台 → 选择公共实例 → 设备管理 → 设备 → 点击查看进入对应的产品 → 点击 MQTT 连接参数查看即可。



注 2: 登录阿里云物联网平台, 进入控制台 → 选择公共实例 → 设备管理 → 产品 → 点击查看进入对应的产品 → Topic 类列表 → 自定义 Topic



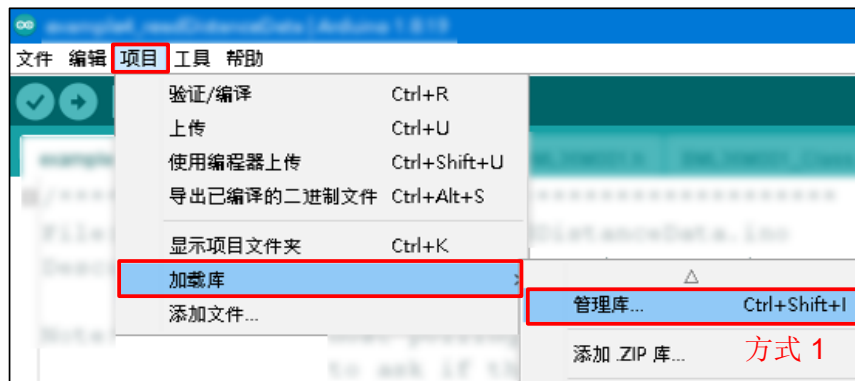
The screenshot shows the 'Topic List' page in the IoT Platform console. The left sidebar contains navigation options: 设备管理, 产品 (selected), 设备, 分组, 设备模拟器, 设备分发, IoT孪生引擎 New, 消息转发, 监控运维, 安全中心, 仿真实验, and 文档与工具. The main content area is titled '智能的开关' with a '发布' button. It displays the ProductKey 'hj3hM0mUhMj' and '1' device count. There are tabs for '产品信息', 'Topic 类列表' (selected), and '功能定义'. Under 'Topic 类列表', there are sub-tabs for '基础通信 Topic', '物模型通信 Topic', and '自定义 Topic'. A '定义 Topic 类' button is present. Below, a list of custom topics is shown: '/hj3hM0mUhMj/\${deviceName}/user/arduino', '/hj3hM0mUhMj/\${deviceName}/user/update' (highlighted with a red box and labeled 'publishstopic'), '/hj3hM0mUhMj/\${deviceName}/user/update/error', and '/hj3hM0mUhMj/\${deviceName}/user/get' (highlighted with a red box and labeled 'subscribetopic').

## Arduino Lib 下载及安装

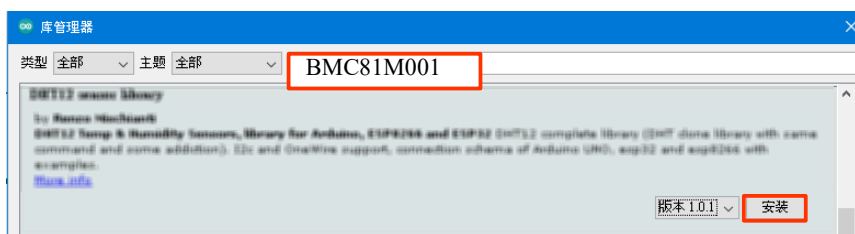
BMC81M001 Library: 可参考下面两种方法安装 BMC81M001 的 Arduino Library

### 方式 1: 搜索安装

搜索安装: Arduino IDE → 项目 → 加载库 → 管理库 → 搜索 BMC81M001 → 安装



搜索安装流程 1

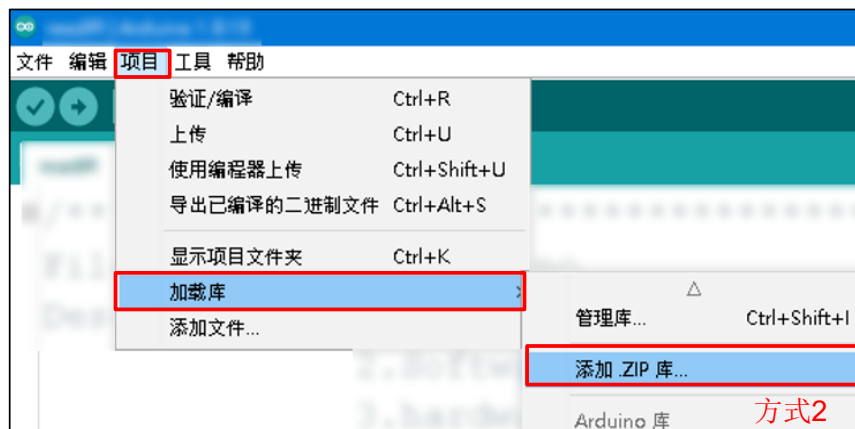


搜索安装流程 2

### 方式 2: 添加 .ZIP 库, 需提前下载 .ZIP 库

下载方法: 打开倍创官方网站 (<https://www.bestmodulescorp.com/bmc81m001.html>) 文件目录下的 Arduino 范例程序 (BMC81M001 Library)。

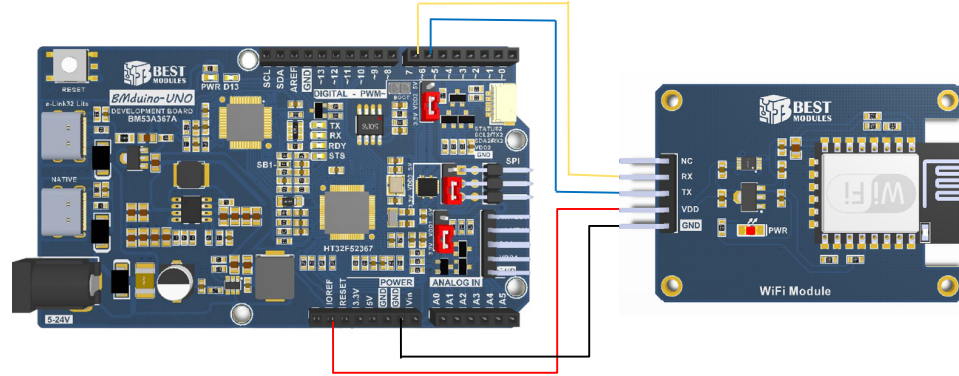
添加 .ZIP 库: Arduino IDE → 项目 → 加载库 → 添加 .ZIP 库 ...





## Arduino 范例

### 范例 1: TCP



实物连接示意图

范例 1 实现功能: 模块连接手机热点, 实现与手机上的 TCP 服务端 APP 双向数据通信。

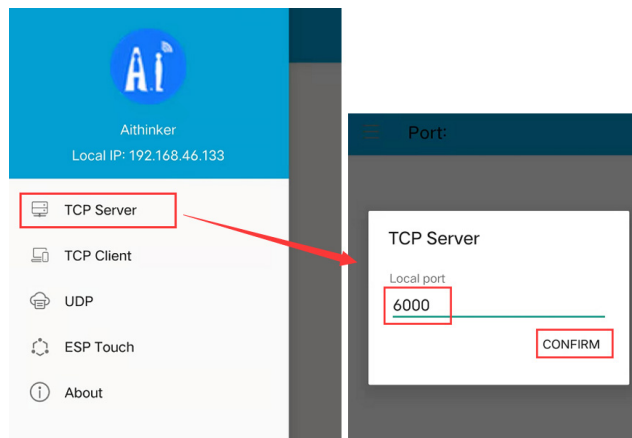
#### 1. 范例打开方式:

Arduino IDE → 文件 → 示例 → Lib 选择 (BMC81M001) → 选择范例 (TCP)

#### 2. 示例说明:

首先需要在 TCP.h 文件中修改 WiFi 信息和 TCP 服务器信息如下:

```
// 连接的 WiFi 的账户密码
#define WIFI_SSID "iQOO_Neo_855" // 手机热点名称
#define WIFI_PASS "12345678.!" // 手机热点密码
// 连接的 TCP IP 及端口信息
#define IP "192.168.46.133" // APP 的 TCP 服务器 IP 地址
#define IP_Port 6000 // APP 的 TCP 服务器端口
```



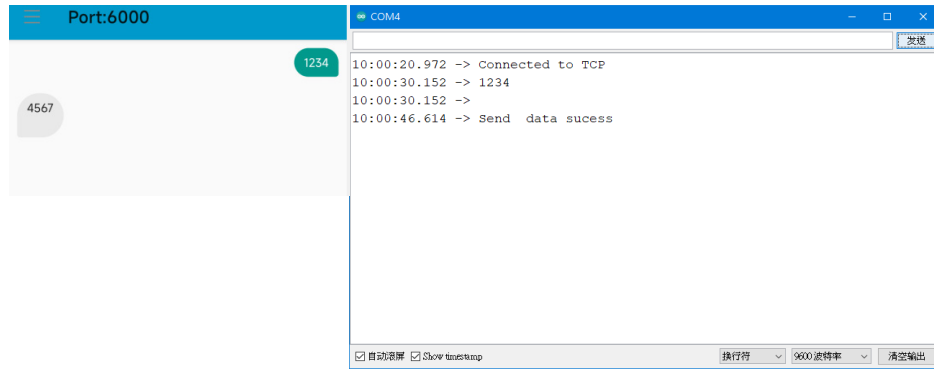
### 3. 连接 WiFi，接入 TCP 服务器

```
#include "TCP.h"
BMC81M001 Wifi (6,7);
void setup()
{digitalWrite(LED, LOW);
  Serial.begin(9600);          // 配置串口监视器
  Wifi.begin();               // 初始化模块
  if(!Wifi.connectToAP(WIFI_SSID,WIFI_PASS)) // 设置连接的热点名称和密码
  {
    Serial.println("Disconnect to WIFI");
  }
  if(!Wifi.connectTCP(IP,IP_Port) // 连接到 TCP 服务器 (APP)
  {
    Serial.println("Disconnect to TCP server");
  }
  else
  {Serial.println("Connected to TCP"); }
}
```

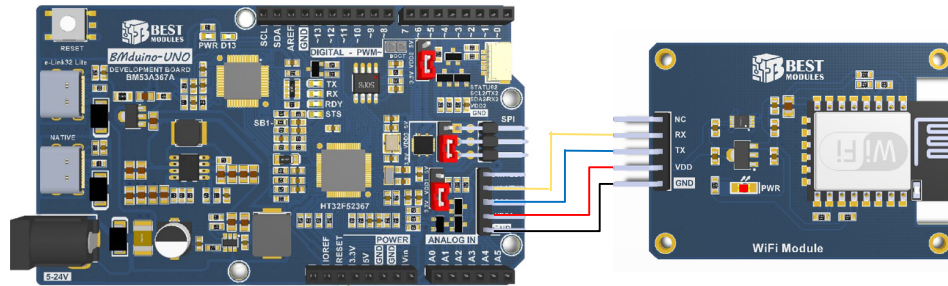
4. 在此范例中，在串口监视器发送数据，将直接上传到 TCP 服务器，在 TCP/UDP 网络调试助手上可以看到模块发送的信息；在 TCP/UDP 网络调试助手上发送的数据可以在串口监视器查看。

```
void loop() {
tcpBuff=Wifi.readDataTcp(); // 监听模块接收的数据
if(tcpBuff!=0)
{
  Serial.println(tcpBuff);
}
while (Serial.available() > 0) // 接收到监视串口发送的数据，透传发送数据
{
  SerialBuff[resLen++] = mSerial.read();
  delay(10);
}
if(resLen>0)
{
  digitalWrite(LED, HIGH); // 有数据亮灯
  if(Wifi. writeDataTcp (resLen,SerialBuff) // 向 TCP 服务器发送数据
  {
    Serial.println("Send data sucess");
    digitalWrite(LED,LOW); // 数据发送成功灭灯
  }
  clearBuff();
}
void clearBuff(){
  memset(SerialBuff, '\0',RES_MAX_LENGTH); // 清空接收串口数据的字符型数组
  resLen = 0;
}
```

程序执行结果在 APP 和串口监视器结果显示：



## 范例 2: Alinyun\_Iot



实物连接示意图

范例 2 实现功能：模块连接手机热点，连接阿里云平台上的实例，实现数据透传功能。

1. 范例打开方式：

Arduino IDE → 文件 → 示例 → Lib 选择 (BMC81M001) → 选择范例 (Alinyun\_Iot)

2. 登录 [物联网平台 \(alinyun.com\)](http://alinyun.com) 注册后选择公共实例，创建产品及设备，自定义 Topic，将 MQTT 参数以及各类 Topic 填入 Alinyun\_Iot.h 文件，具体流程如下：

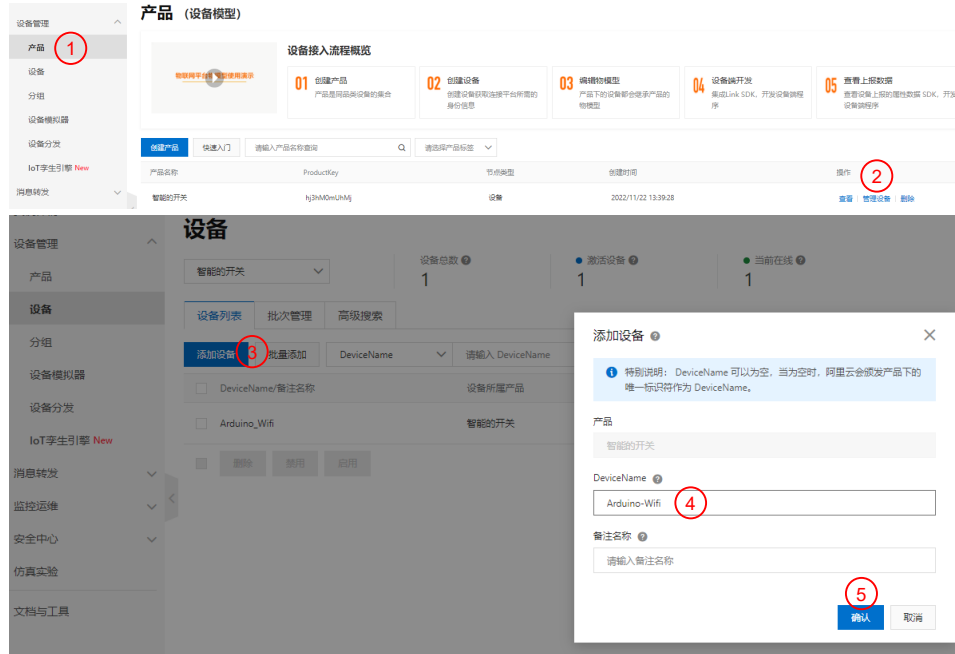
1) 创建产品：进入公共实例后，左侧菜单设备管理 → 产品 → 创建产品



2) 自定义 Topic，定义数据传输中介：设备管理 → 产品 → 查看刚刚新建的产品 → Topic 类列表 → 自定义 Topic



3) 回到“产品”界面 → “管理设备” → “添加设备” → DeviceName, 点击确认。



4) 查看设备的 MQTT 连接参数：设备 → 查看刚刚创建的设备 Arduino-Wifi → 设备信息 → MQTT 参数查看；查看设备的 Topic 同步步骤 2 中的自定义 Topic 步骤相同。

将 CLIENTLID、USERNAME、PASSWORD、MQTT\_HOST、SERVER\_PORT 等 MQTT 参数填入 Aliyun\_Iot.h 文件里，将 PUBLISHTOPIC、SUBSCRIBERTOPIC、CUSTOMTOPIC 等类 Topic 也填入 Aliyun\_Iot.h 文件里。



```

1 #ifndef _BMC81M001_H_
2 #define _BMC81M001_H_
3
4 //***** Header *****//
5 //***** Header *****//
6 //***** Header *****//
7 #include "Aliyun_Iot.h"
8 #include "BMC81M001.h"
9
10 //***** wifi information *****//
11 //***** wifi information *****//
12 //***** wifi information *****//
13
14 #define WIFI_SSID "iQOO_Neo_855"
15 #define WIFI_PASS "12345678.!"
16
17 #define CLIENTLID "mytest|securemode=3\\,signmethod=hr
18 #define USERNAME "Arduino_Wifi&gqzn81RWZC2"
19 #define PASSWORD "9CF1D3420F07ECC02250EF829D9EAC8529A8D672"
20 #define MQTT_HOST "gqzn81RWZC2.iot-as-mqtt.cn-shanghai.aliyuncs.com"
21 #define SERVER_PORT 1883
22
23 #define PUBLISHTOPIC "gqzn81RWZC2/Arduino_Wifi/user/update"
24 #define SUBSCRIBERTOPIC "gqzn81RWZC2/Arduino_Wifi/user/get"
25 #define CUSTOMTOPIC "gqzn81RWZC2/Arduino_Wifi/user/ardunio"
26

```

**MQTT 连接参数**

clientId	hj3hM0mUhmJ/Arduino_Wifi/securemode=2,signmethod=hmacha256,timestamp=167653534057
username	Arduino_Wifi&hj3hM0mUhmJ
passwd	0665e145e05a07cc63274d66766db4e4aaa47d5378ba4c1350f5de67c37670
mqttHostUrl	iot-06z00ac1cwRkn1s.mqtt.iothub.aliyuncs.com
port	1883

**智能的开关** 发布

ProductKey hj3hM0mUhmJ 复制

设备数 1 前往管理

产品信息 Topic 列表 功能定义

基础通信 Topic 物模型通信 Topic

定义 Topic 类

自定义 Topic

- /hj3hM0mUhmJ/\${deviceName}/user/ardunio **customtopic**
- /hj3hM0mUhmJ/\${deviceName}/user/update **publishtopic**
- /hj3hM0mUhmJ/\${deviceName}/user/update/error
- /hj3hM0mUhmJ/\${deviceName}/user/get **subscribetopic**

```

// 连接的 WiFi 的账户密码
#define WIFI_SSID "iQOO_Neo_855" // 手机热点名称
#define WIFI_PASS "12345678.!" // 手机热点密码
// aliyun MQTT 信息
#define CLIENTLID "mytest|securemode=3\\,signmethod=hmacha256\\,timestamp=6789|"
#define USERNAME "Arduino_Wifi&gqzn81RWZC2"
#define PASSWORD "9CF1D3420F07ECC02250EF829D9EAC8529A8D672"
#define MQTT_HOST "gqzn81RWZC2.iot-as-mqtt.cn-shanghai.aliyuncs.com"
#define SERVER_PORT 1883 // 端口号
// aliyun TOPIC 信息
#define PUBLISHTOPIC "gqzn81RWZC2/Arduino_Wifi/user/update"
#define SUBSCRIBERTOPIC "gqzn81RWZC2/Arduino_Wifi/user/get"
#define CUSTOMTOPIC "gqzn81RWZC2/Arduino_Wifi/user/ardunio"

```

**注意：** CLIENTID 填写时需留意在 “,” 前需加 \\。

### 3. 示例说明:

根据阿里云平台产生的数据，对模块进行初始化。其中包括软件复位模块，接入 WiFi，接入阿里云 IoT，如下：

```
#include "Aliyun_Iot.h"
BMC81M001      Wifi(&Serial1);
void setup()
{
  digitalWrite(LED, LOW);
  Serial.begin(9600);           // 配置串口监视器
  Wifi.begin();                // 模块初始化及配置
  Wifi.reset();
  Serial.print("WIFI Connection Results: ");
  if(Wifi.connectToAP(WIFI_SSID,WIFI_PASS)==0) // 根据名称和密码连接 WiFi
  {
    Serial.println("fail");
  }
  else {Serial.println("success");}
  Serial.print("Aliyun Connection Results: ");
  Wifi.sendATCommand("AT+CIPSPNTPCFG=1,8,\"ntp1.aliyun.com\"",1000,2);
  // 连接阿里云
  if(Wifi.configMqtt(CLIENTLID,USERNAME,PASSWORD,MQTT_HOST,
    SERVER_PORT)==0) // 配置MQTT参数
  {
    Serial.println("fail");
  }
  else
  {
    Serial.println("success");
    Wifi.setPublishTopic(PUBLISHTOPIC); // 连接成功后配置订阅和发布 Topic
    Wifi.setSubscribetopic(SUBSCRIBERTOPIC);
  }
  Serial.print("Topic set Results: ");
  if(Wifi.setTopic(CUSTOMTOPIC)==0) // 设置自定义 Topic
  {
    Serial.println("fail");
  }
  else {Serial.println("success");}
  delay(200);}
}
```

### 4. 运行后，即可在平台上查看到设备状态。

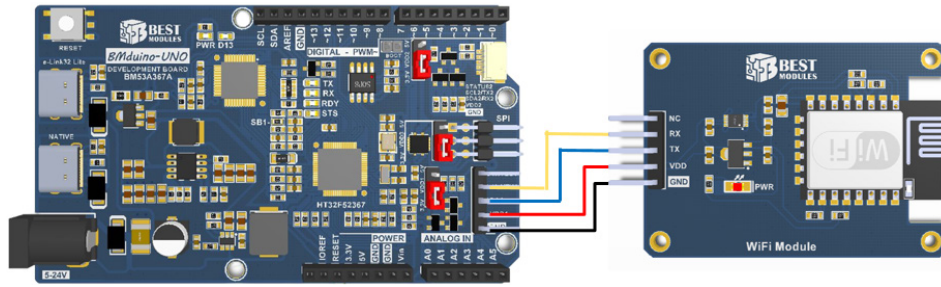
<input type="checkbox"/> DeviceName/备注名称	设备所属产品	节点类型	设备状态 	最后上线时间	启用/禁用	操作
<input type="checkbox"/> Arduino_Wifi	智能的开关	设备	<span style="color: green;">●</span> 在线	2023/02/16 15:59:01.886	<input checked="" type="checkbox"/>	<a href="#">查看</a> <a href="#">删除</a>

5. 在此范例中，当串口监视器发送数据，模块只做透传功能，将接收到串口监视器发送的数据直接上传到云平台，可以在平台日志记录查看到模块上传的信息。

```
void loop()
{
  Wifi.readIotData(&aliyunReciveBuff, &aliyunReciveBufflen, &recTopic);
  // 监听模块接收的数据
  if(aliyunReciveBufflen)
  {
    Serial.println(aliyunReciveBufflen);
    Serial.println(aliyunReciveBuff);
  }
  // 接收到监视串口发送的数据，透传发送数据
  while (Serial.available() > 0) // 接收监视串口发送的数据
  {
    SerialBuff[resLen++] = Serial.read();
    delay(10);
  }
  if(resLen>0)
  {
    digitalWrite(LED, HIGH);
    DATA_BUF = (String )SerialBuff;
    topic = PUBLISHTOPIC;
    if(Wifi.writeString(DATA_BUF,topic))
    {
      Serial.println("Send String data sucess");
    }
    if(Wifi.writeBytes(SerialBuff, resLen,topic))
    {
      Serial.println("Send byte data sucess");
    }
    clearBuff();
  }
}
void clearBuff() {
  memset(SerialBuff, '\0', RES_MAX_LENGTH); // 清空数据接收数组
  resLen = 0;
}
```



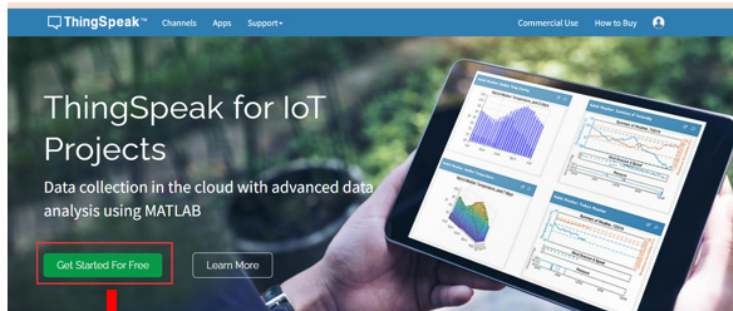
### 范例 3: ThingSpeak



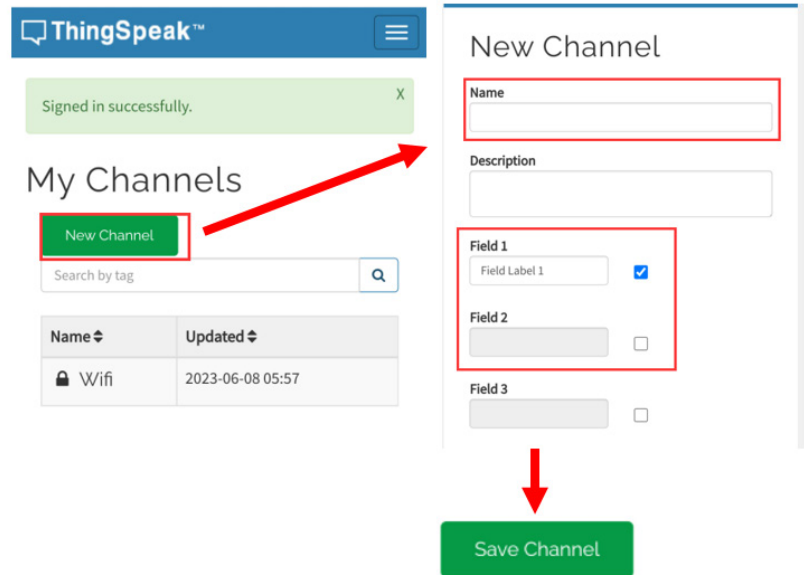
实物连接示意图

范例 3 实现功能: 模块连接手机热点, 连接 ThingSpeak 平台上的通道, 发送数据到平台上显示。

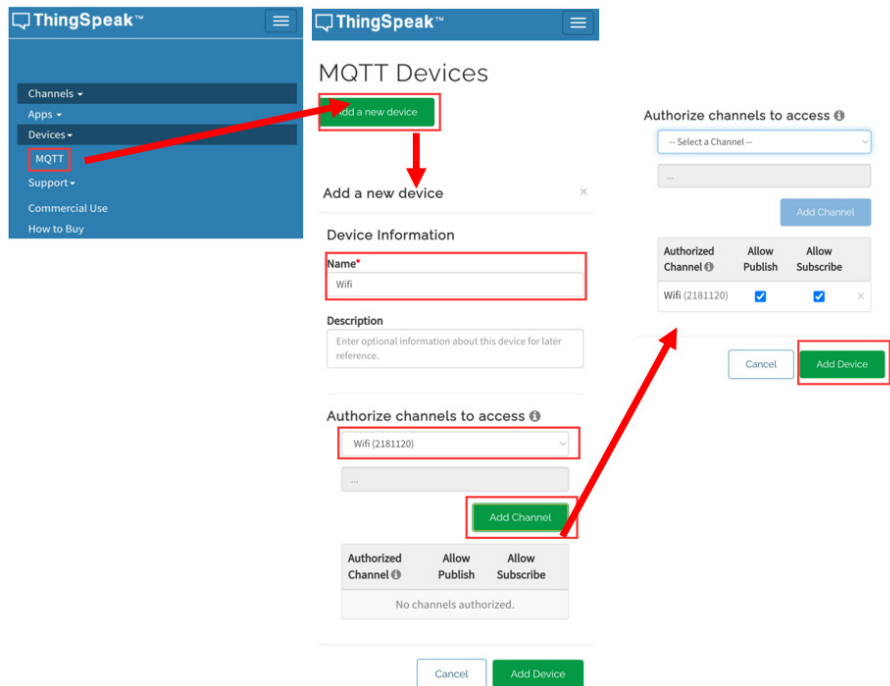
1. 范例打开方式: Arduino IDE → 文件 → 示例 → Lib 选择 (BMC81M001) → 选择范例 (ThingSpeak)
2. 登录 ThingSpeak 平台 (<https://thingspeak.com/>), 注册账号后, 新建数据通道, 再新建 MQTT 设备, 将 MQTT 参数填入 ThingSpeak.h 文件中。
  - 1) 注册账号并进入平台: 点击 Get Start For Free, 进入平台界面, 注册账号并登入。



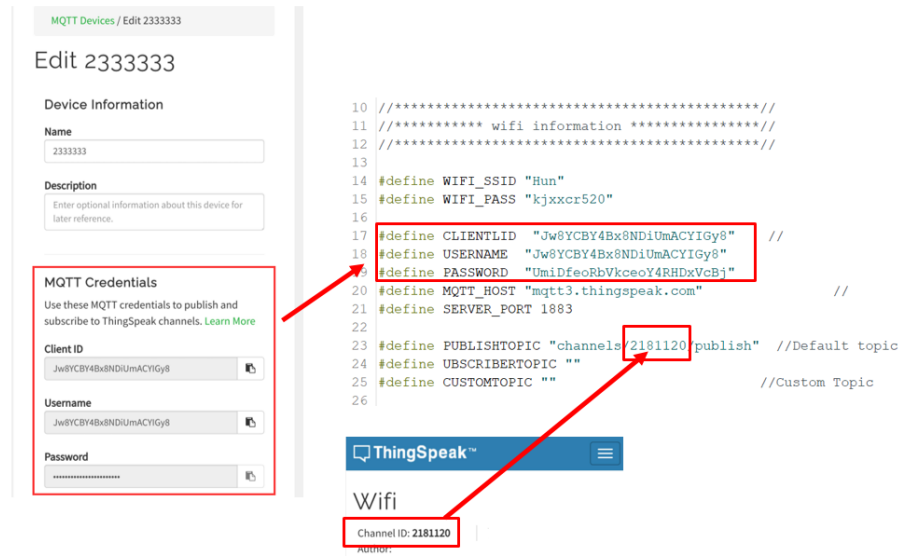
2) 创建 channels: 点击 New Channel, 填入 Name 和开启数据字段数量, 保存通道即可。



3) 新建 MQTT 设备: 填入设备名称 Wifi, 选择新建的通道 Wifi, 点击 Add channel, 显示 Allow Publish 和 Sublish OK 后 Add Device。



4) 点击 edit, 查看新建 MQTT 设备的 MQTT 参数, 将数据填入 ThingSpeak.h 文件中。



The screenshot shows the 'Edit 2333333' page for an MQTT device. The 'MQTT Credentials' section is highlighted with a red box. The code on the right shows the corresponding definitions in ThingSpeak.h. Red boxes and arrows indicate the mapping: Client ID, Username, Password, and Publish Topic.

```

// 要连接的热点信息
#define WIFI_SSID "iQ00_Neo_855" // 手机热点名称
#define WIFI_PASS "12345678.!" // 手机热点密码
// ThingSpeak MQTT 信息
#define CLIENTLID "Jw8YCBY4Bx8NDiUmACYIGy8"
#define USERNAME "Jw8YCBY4Bx8NDiUmACYIGy8"
#define PASSWORD "UmiDfeoRbVkceoY4RHDxVcBj"
#define MQTT_HOST "mqtt3.thingspeak.com" // 服务器地址
#define SERVER_PORT 1883 // 端口号
// ThingSpeak TOPIC 信息
#define PUBLISH_TOPIC "channels/2181120/publish"
    
```

### 3. 示例说明:

模块初始化后，连接热点，配置 MQTT 参数，连接 ThingSpeak 平台的通道，发送数据至平台。如下：

```
#include "ThingSpeak.h"
BMC81M001      Wifi(&Serial1);
void setup()
{
  digitalWrite(LED, LOW);
  Serial.begin(9600);           // 配置串口监视器
  Wifi.begin();                // 模块初始化及配置
  Wifi.reset();
  Serial.print("WIFI Connection Results: ");
  if(Wifi.connectToAP(WIFI_SSID,WIFI_PASS)==0) // 根据名称和密码连接 WiFi
  {
    Serial.println("fail");
  }
  else {Serial.println("success");}
  Serial.print("ThingSpeak Connection Results: ");
  if(Wifi.configMqtt (CLIENTID,USERNAME,PASSWORD,MQTT_HOST,
    SERVER_PORT)==0)          // 配置 MQTT 参数
  {
    Serial.println("fail");
  }
  else
  {
    Serial.println("success");
  }
  delay(200);}

```

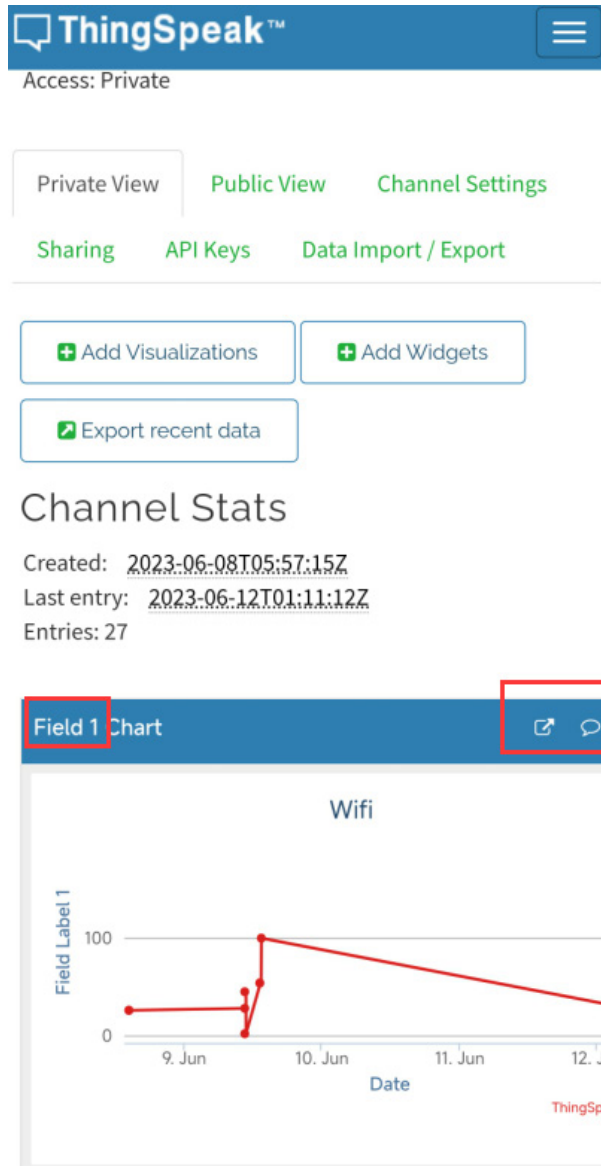
4. 在此范例中，在串口监视器发送数据，数据会直接上传到云平台，可以在通道接口看到数据。

```
void loop()
{
  Wifi.readIotData (&RecvBuff, &RecvBuffLen, &recTopic); // 监听模块
                                                         // 接收的数据

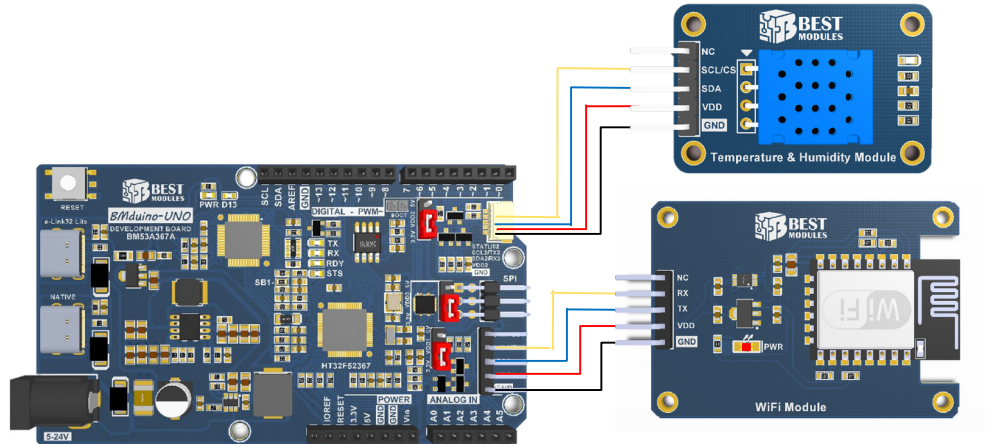
  if (RecvBuffLen)
  {
    Serial.println(RecvBuffLen);
    Serial.println(RecvBuff);
  } // 接收到监视串口发送的数据，透传发送数据
  while (Serial.available() > 0) // 接收监视串口发送的数据
  {
    SerialBuff[resLen++] = Serial.read();
    delay(10);
  }
  if (resLen > 0)
  {
    digitalWrite(LED, HIGH);
    DATA_BUF = "field1="; // 发送通道字段 1 数据
    DATA_BUF += SerialBuff;
    topic = PUBLISHTOPIC;
    if (Wifi.writeString(DATA_BUF, topic))
    {
      Serial.println("Send String data sucess");
    }
    clearBuff();
    digitalWrite(LED, LOW);
  }
}

void clearBuff() {
  memset(SerialBuff, '\0', RES_MAX_LENGTH); // 清空数据接收数组
  resLen = 0;
}
```

5. ThingSpeak 显示结果，可以从图表中看到数据的接收。



## 范例 4: ThingSpeakPublish

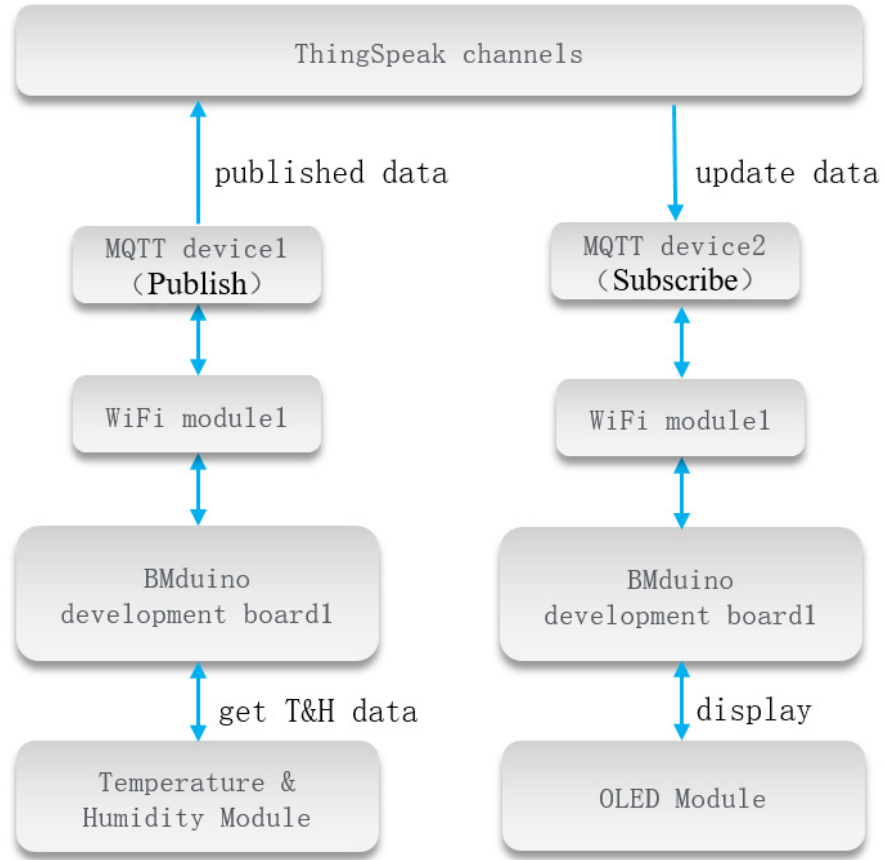


实物连接示意图

范例 4 和范例 5 可搭配使用，实现两个 WiFi 设备的远程数据交换，一个作为 Publish 端，另一个作为 Subscribe 端。实现功能如下：

**Publish 端：**将温湿度模块 BME33M251 的温度、湿度数据上传到 ThingSpeak；

**Subscribe 端：**从 ThingSpeak 读取 Publish 端刚上传的温度、湿度数据，并显示于 OLED 模块上；

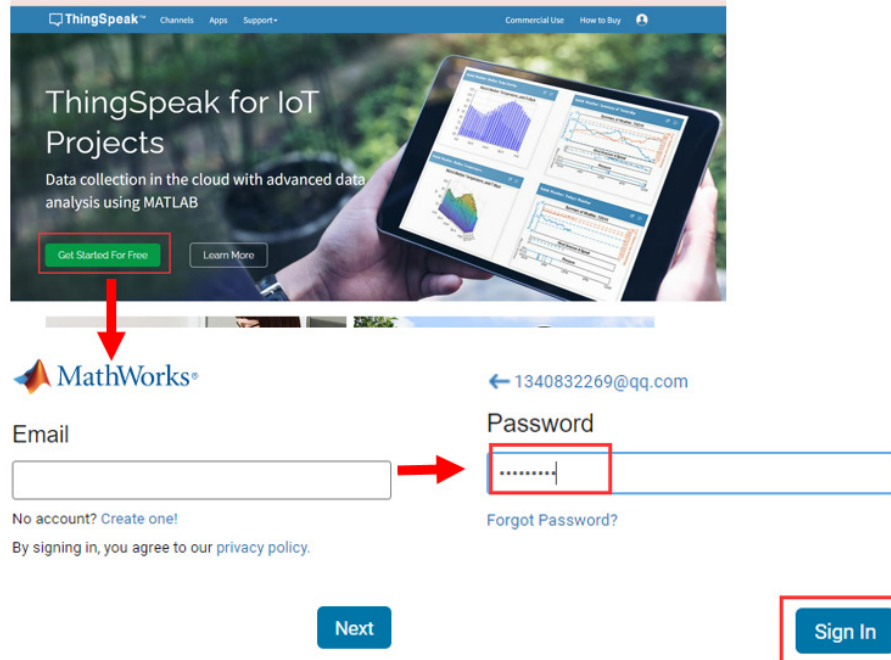


范例 4 实现功能：读取温湿度模块 BME33M251 的温度、湿度数据，通过 MQTT 设备 1（Publish）发布到 ThingSpeak 平台的通道中。

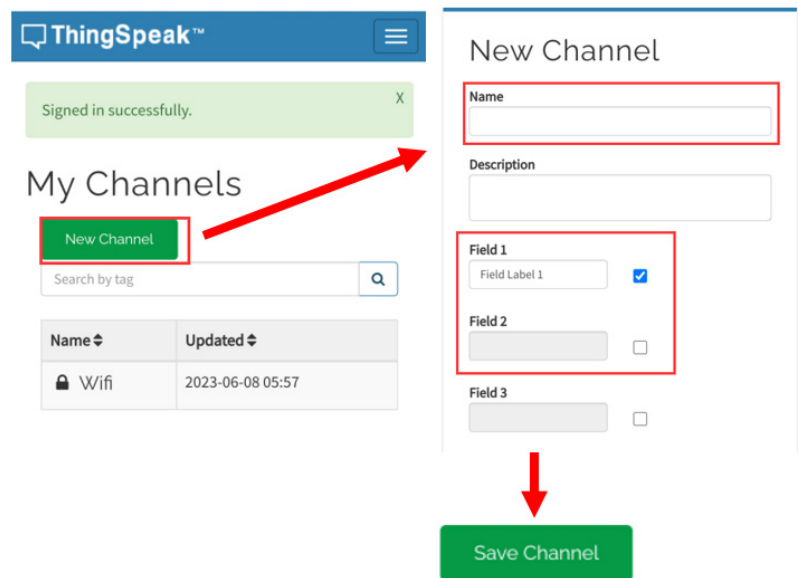
1. 范例打开方式：Arduino IDE → 文件 → 范例 → Lib 选择 (BMC81M001) → 选择范例 (ThingSpeakPublish)。
2. 登录 ThingSpeak 平台 (<https://thingspeak.com>)，注册账号后，新建数据通道 (wifi t&h)，新建 MQTT 设备 (Publish)，将 MQTT 参数填入 ThingSpeakPublish.h 文件中。



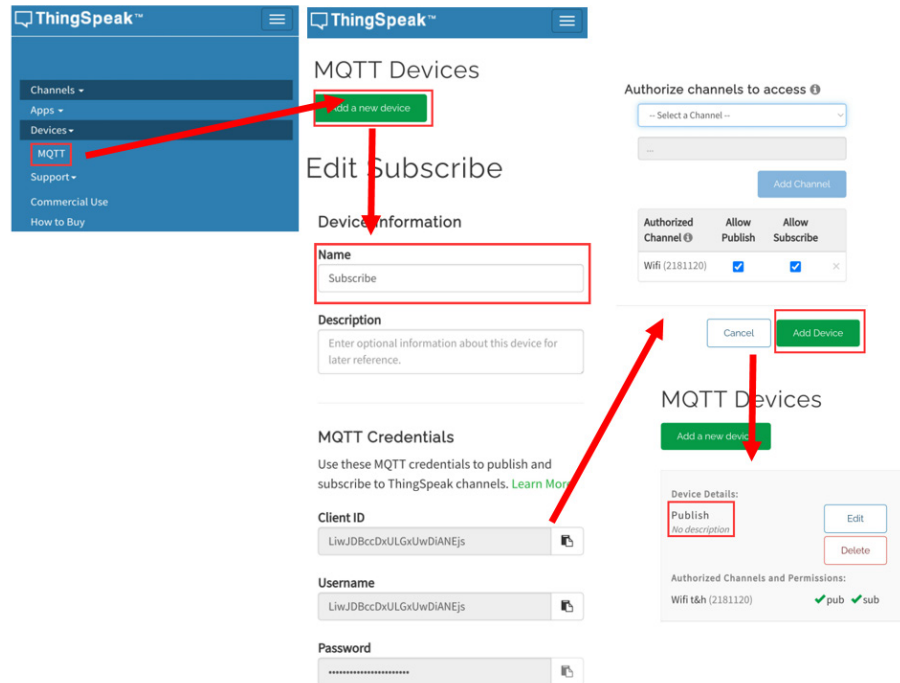
- 1) 注册账号并进入平台：点击 Get Start For Free，进入平台界面，注册账号并登入。



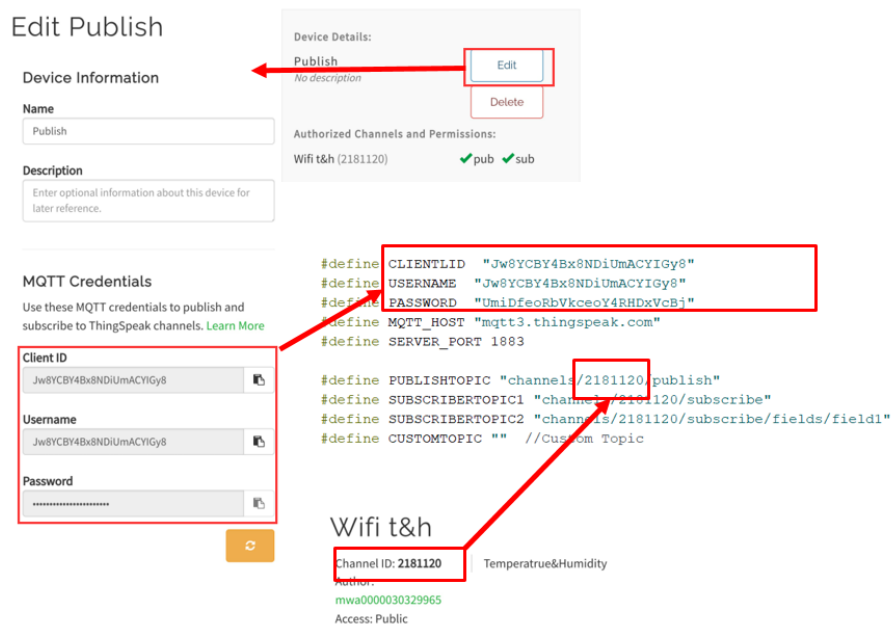
- 2) 创建 channels: 点击 New Channel，填入自定义 Name (例 wifi t&h) 和 Field1 (例 humidity) 和 Field2 (例 temperature)，保存通道即可。



3) 新建 MQTT 设备 1: 填入设备名称 (Publish), 授权通道 wifi t&h 的发布和订阅, 点击 Add channel, 显示 Allow Publish 和 Sublish OK 后 Add Device。此 Publish 设备用作上传温湿度模块数据。



4) 点击设备 (Publish) 右侧的“edit”按钮, 查看 MQTT 设备 (Publish) 的 MQTT 参数, 将数据填入 ThingSpeakPublish.h 文件中。



### 3. 示例说明:

模块初始化后, 连接热点, 配置 MQTT 参数, 连接 ThingSpeak 平台的通道, 如下:

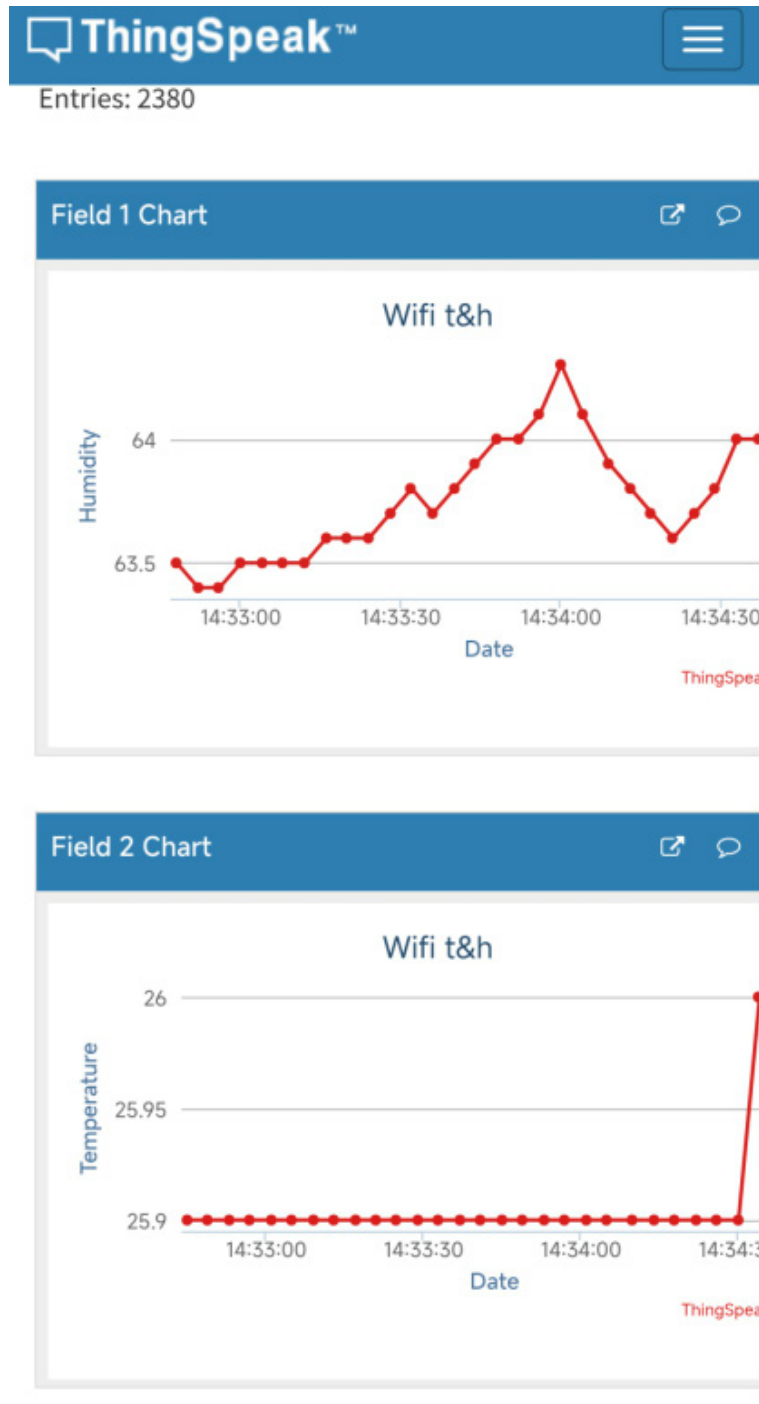
```
#include "ThingSpeak.h"
#include <BMD31M090.h>
#include <BM25S2021-1.h> // 温湿度模块
BMC81M001 Wifi(&Serial1);
BM25S2021_1 BMht(&Wire2);
void setup()
{
  Serial.begin(9600);
  BMht.begin();
  Wifi.begin(); //WIFI 模块、温湿度模块初始化
  Wifi.reset();
  Serial.print("WIFI Connection Results :");
  if(Wifi.connectToAP(WIFI_SSID,WIFI_PASS)==0) // 连接热点
  {
    Serial.println("fail");
  }
  else {Serial.println("success");}
  Serial.print("ThingSpeak Connection Results :");
  if(Wifi.configMqtt(CLIENTLID,USERNAME,PASSWORD,MQTT_HOST,SERVER_PORT)==0)
  // 根据 MQTT 连接 ThingSpeak 平台
  {
    Serial.println("fail");
  }
  else {Serial.println("success");}
  delay(200);
  Wifi.setPublishTopic(PUBLISHTOPIC); // 订阅 Topic · 即通道数据更新
  Wifi.setSubscribetopic(SUBSCRIBERTOPIC2);
  Wifi.setSubscribetopic(SUBSCRIBERTOPIC1);
  topic = PUBLISHTOPIC; // 发布 Topic
}
```

4. 轮询温湿度模块的数据，将湿度发送至通道中的 field1，将温度发送至通道的 field2。

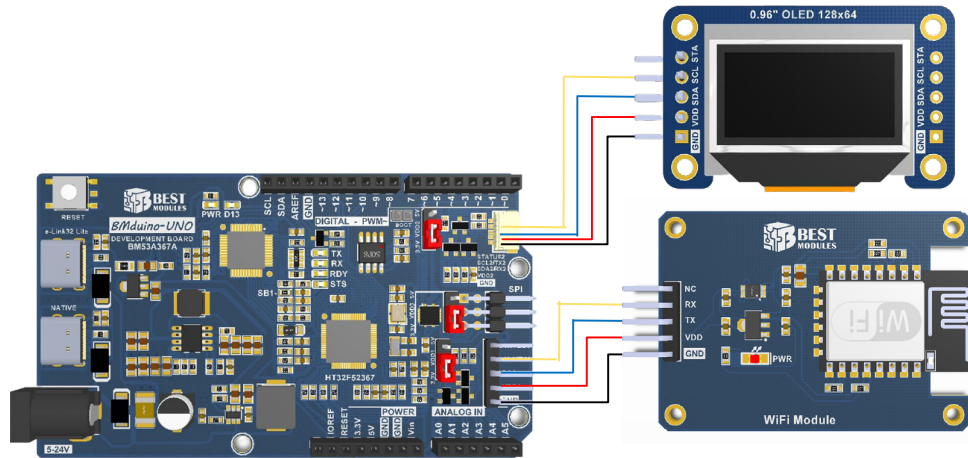
```
void loop()
{
  data1=BMht.readHumidity();           // 获取湿度数据
  Humidity=String(data1,2);           // 浮点型数据转成字符串
  DATA_BUF = "field1=";              // 通道 1 数据
  DATA_BUF += Humidity;
  if(Wifi.writeString(DATA_BUF,topic) // 发布数据至通道 1
    {
      Serial.println("Send String data sucess");
      delay(1000);
    }
  clearBuff();

  data1=BMht.readTemperature(false); // 获取温度数据
  Temperature=String(data1,2);       // 浮点型数据转成字符串
  DATA_BUF = "field2=";              // 通道 2 数据
  DATA_BUF += Temperature;
  if(Wifi.writeString(DATA_BUF,topic) // 发布数据至通道 2
    {
      Serial.println( "Send String data sucess" );
      delay(1000);
    }
  clearBuff();
  delay(2000);
}
```

5. ThingSpeak 显示结果，可以从图表中看到数据的接收。



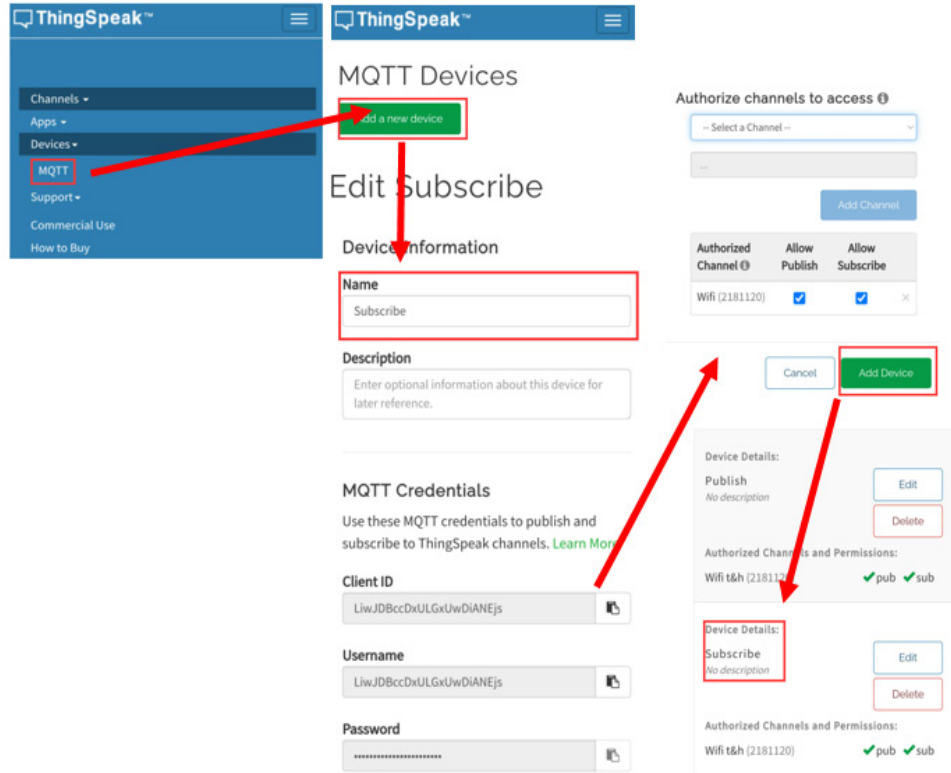
## 范例 5: ThingSpeakSubscribe



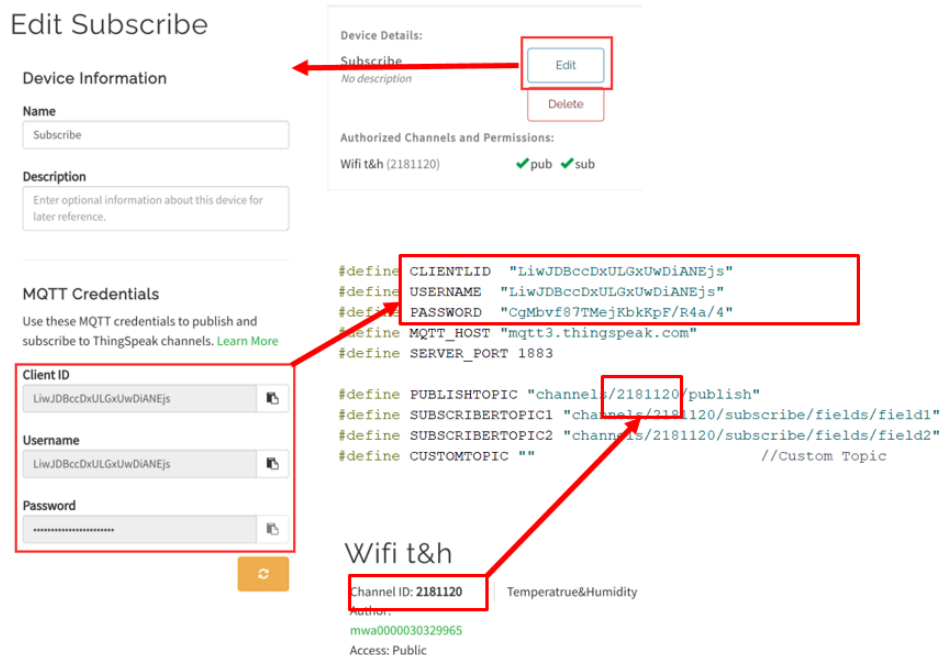
实物连接示意图

范例 5 实现功能：订阅数据更新 Topic，通过新建并连接 MQTT 设备 2（Subscribe）接收 ThingSpeak 对应通道最新发布的高温湿度数据，并显示在 OLED 模块上。

1. 范例打开方式：Arduino IDE → 文件 → 范例 → Lib 选择 (BMC81M001) → 选择范例 (ThingSpeakSubscribe)。
2. ThingSpeak 通道无需新建，使用范例 4 的 wifi t&h。下面新建 MQTT 设备 (Subscribe) 以及填写参数至 ThingSpeakSubscribe.h 文件。
  - 1) 新建 MQTT 设备 2：填入设备名称 (Subscribe)，授权通道 wifi t&h 的发布和订阅，点击 Add channel，显示 Allow Publish 和 Sublish OK 后 Add Device。此 Subscribe 设备用于接收通道最新更新的数据。



4) 点击设备 (Subscribe) 右侧的“edit”按钮，查看 MQTT 设备 (Subscribe) 的 MQTT 参数，将数据填入 ThingSpeakSubscribe.h 文件中。



### 3. 示例说明:

模块初始化后, 连接热点, 配置 MQTT 参数, 连接 ThingSpeak 平台的通道。  
如下:

```
#include "ThingSpeakRead.h"
#include <BMD31M090.h>
#include <BM25S2021-1.h> //OLED 模块
BMC81M001 Wifi(&Serial1);
BMD31M090 BMD31(128,64, &Wire2);
float data1;
String Humidity;
String Temperature;
void Display();
void setup()
{
  BMD31.begin(0x3C);
  BMD31.setFont(FontTable_8X16);
  delay(100);
  Serial.begin(9600);
  Wifi.begin();
  Wifi.reset();
  Display();
  Serial.print("WIFI Connection Results :");
  if(Wifi.connectToAP(WIFI_SSID,WIFI_PASS)==0) // 连接热点
  {
    Serial.println("fail");
  }
  else {Serial.println("success");}
  Serial.print("ThingSpeak Connection Results :");
  if(Wifi.configMqtt(CLIENTLID,USERNAME,PASSWORD,MQTT_HOST,SERVER_PORT)==0)
  // 根据 MQTT 连接 ThingSpeak 平台
  {
    Serial.println("fail");
  }
  else {Serial.println("success");}
  delay(200);
  Wifi.setPublishTopic(PUBLISHTOPIC); // 订阅 Topic · 即通道数据更新
  Wifi.setSubscribetopic(SUBSCRIBERTOPIC2);
  Wifi.setSubscribetopic(SUBSCRIBERTOPIC1);
}
```



4. 轮询平台是否有更新数据，有数据则识别对应数据，显示在 OLED 模块上。

```
void loop()
{
  Wifi.readIotData (&ReciveBuff, &ReciveBufflen, &topic); // 接收平台下发的数据
  if (ReciveBufflen)
  {
    if (topic==SUBSCRIBERTOPIC1) // 通道 1 数据为湿度
    {
      ReciveBuff.toCharArray (OledBuff, 7);
      BMD31.drawString (40, 2, (u8*) OledBuff); // OLED 显示
      BMD31.drawString (85, 2, (u8*) "%");
      ReciveBufflen=0;
    }
    if (topic==SUBSCRIBERTOPIC2) // 通道 2 数据为温度
    {
      ReciveBuff.toCharArray (OledBuff, 7);
      BMD31.drawString (40, 6, (u8*) OledBuff); // OLED 显示
      BMD31.drawString (85, 6, (u8*) "C");
      ReciveBufflen=0;
    }
  }
  clearBuff (); // 清空数据
}
```

5. OLED 显示结果如下：



Copyright© 2023 by BEST MODULES CORP. All Rights Reserved.

本文件出版时倍创已针对所载信息为合理注意，但不保证信息准确无误。文中提到的信息仅是提供作为参考，且可能被更新取代。倍创不承担任何明示、默示或法定的，包括但不限于适合商品化、令人满意的质量、规格、特性、功能与特定用途、不侵害第三方权利等保证责任。倍创就文中提到的信息及该信息之应用，不承担任何法律责任。此外，倍创并不推荐将倍创的产品使用在会由于故障或其他原因而可能会对人身安全造成危害的地方。倍创特此声明，不授权将产品使用于救生、维生或安全关键零部件。在救生 / 维生或安全应用中使用倍创产品的风险完全由买方承担，如因该等使用导致倍创遭受损害、索赔、诉讼或产生费用，买方同意出面进行辩护、赔偿并使倍创免受损害。倍创 ( 及其授权方，如适用 ) 拥有本文件所提供信息 ( 包括但不限于内容、数据、示例、材料、图形、商标 ) 的知识产权，且该信息受著作权法和其他知识产权法的保护。倍创在此并未明示或暗示授予任何知识产权。倍创拥有不事先通知而修改本文件所载信息的权利。如欲取得最新的信息，请与我们联系。