

MicroPython for OneNET

物联网入门套件
使用手册



由芝麻 DIY 汇编

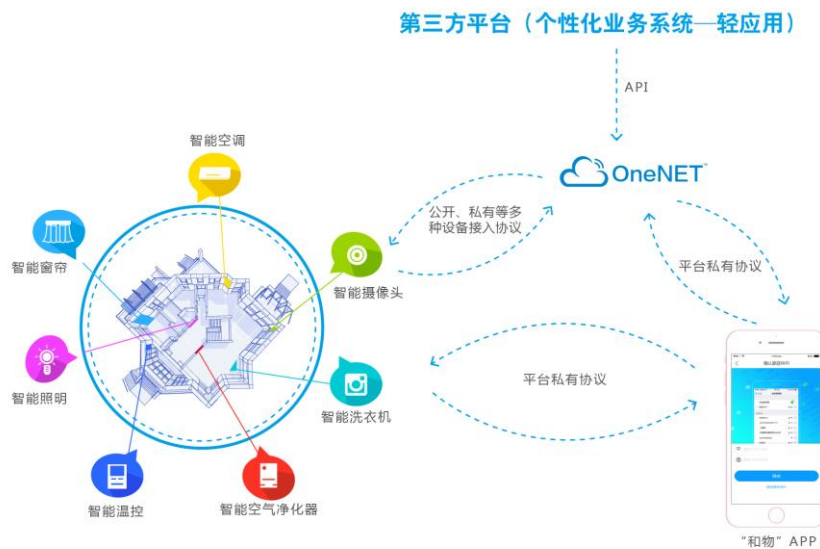
目录

一、OneNET 简介.....	3
二、MQTT 简介	4
三、需要的电子学知识	6
1. 面包板	6
2. 电源	7
3. 模拟和数字 I/O.....	7
4. 上拉和下拉.....	10
5. 信号消抖.....	11
四、套件各个模块介绍	14
1. 面包板	14
2. 杜邦线	15
3. 蜂鸣器模块.....	15
4. 电阻器	16
5. 发光二极管.....	16
6. 轻触开关.....	17
7. 门磁开关.....	17
五、注册 OneNET 平台	19
六、例程解析	25
1. 平台控制 LED	26
2. 向平台上传静态数据.....	29
3. 控制蜂鸣器.....	31
4. 上传 DHT11 温湿度数据.....	32
5. 门磁开关.....	35
6. 设备间通信.....	38

注：由于编者水平有限，文中难免错误和不妥，望广大读者批评指正

一、OneNET 简介

中国移动物联网开放平台是中移物联网有限公司基于物联网技术和产业特点打造的开放平台和生态环境，适配各种网络环境和协议类型，支持各类传感器和智能硬件的快速接入和大数据服务，提供丰富的 API 和应用模板以支持各类行业应用和智能硬件的开发，能够有效降低物联网应用开发和部署成本，满足物联网领域设备连接、协议适配、数据存储、数据安全、大数据分析等平台级服务需求。

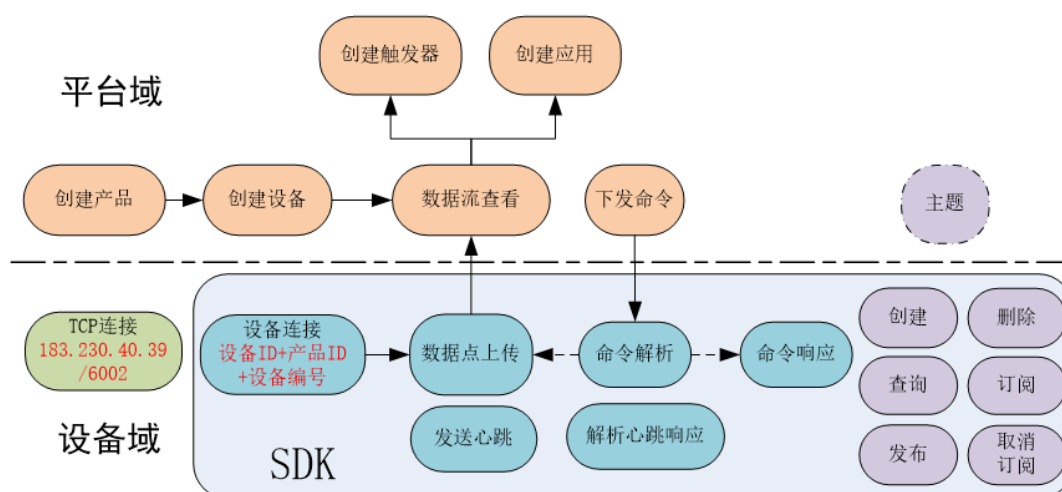


中国移动物联网开放平台始终秉承开放合作的态度，为智能硬件创客和创业企业提供硬件社区服务，为中小企业客户物联网应用需求提供数据展现、数据分析和应用生成服务，为重点行业领域/大客户提供行业 PaaS 服务和定制化开发服务。

OneNET 聚焦各大行业痛点需求，在智能家居、智慧车载、智慧穿戴、智慧能源以及工业制造等行业提供完整的解决方案。

二、MQTT 简介

MQTT (Message Queuing Telemetry Transport, 消息队列遥测传输) 是 IBM 开发的一个即时通讯协议, 有可能成为物联网的重要组成部分。该协议支持所有平台, 几乎可以把所有联网物品和外部连接起来, 被用来当做传感器和致动器的通信协议, 采用轻量级发布和订阅消息传输机制, 拥有转换成本低, 网络要求低, 硬件成本低, 安全性高等诸多优点。



转换成本低

基于 TCP\IP 的出身, 工程师们几乎不需要改造目前广泛使用以太网链路, 就可以直接部署实施。

网络要求低

物联网不比移动互联网, 联网设备形态各异、网络状况未必理想, 容错率低。针对这些状况, MQTT 采取遗言机制巧妙应对。针对物联网, 众多协议中只有 MQTT 完美照顾了嵌入式设备, 最小的数据包仅有 2 个比特!

硬件要求低

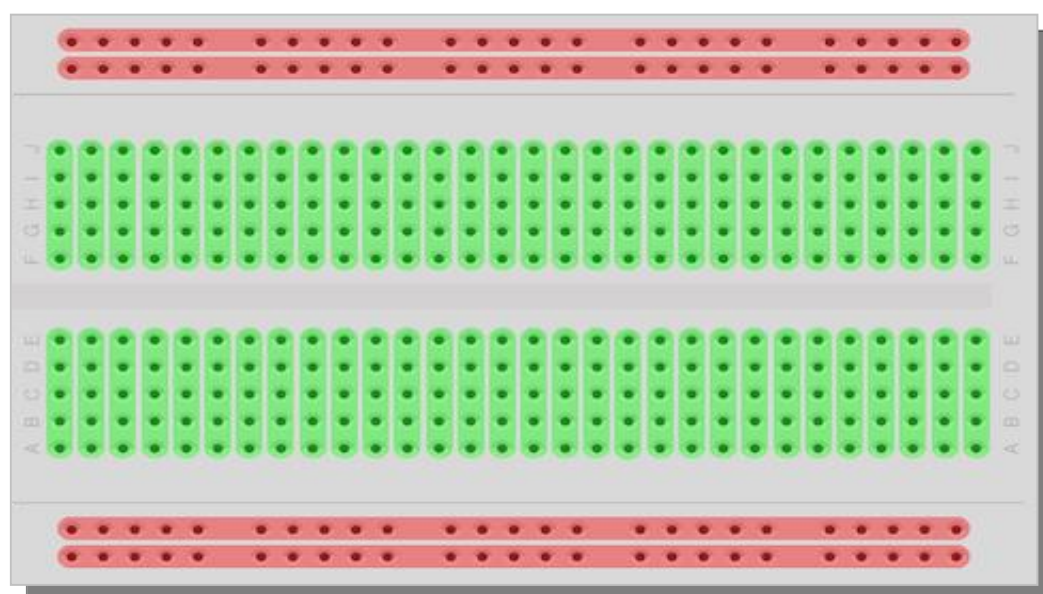
设备接入网络后，依赖网络稳定或者设备的运算存储工作依然是移动互联网的打法。物联网的接入设备应当是轻量的、低成本的、可快速铺开形成大数据效应和服务的设备。一个 MQTT 的 Client 端程序经常只有几十 k 大小。这就意味着更低的能耗，和更稳定的接入。

三、需要的电子学知识

本篇提供一些基础的电子学概念，给一些没有电子基础的同学，你应该了解这些，这样可以确保你可靠地连接物理设备，并正确的编写程序以使整个项目可以运转起来。在你真正动手连接设备之前，你应大概的了解电压，电流，电阻值，以及电源之间的关系，如果想更深一步的了解，可以去寻找“电路基础”和“欧姆定律”相关的知识。

1.面包板

面包板是用来快速的做一些电子实验的工具，使用面包板做电子实验可以避免使用烙铁和焊锡，简单方便并且安全，在使用时只需要将电子元器件插到面包板上，然后使用杜邦线将元器件各引脚连接起来就可以了。



如上图所示，面包板上有很多小孔，有一些孔在内部是连接在一起的，这样只要不同的元器件的引脚在同一排或同一列上，就可以使他们的引脚连接起来，一般的面包板就像图中那样，中间部分是竖着连接起来的，两边的部分是横着连接起来的。

2.电源

嵌入式设备都包含有源电路，这意味着设备需要一个外置的电源，像充电器，电池，USB 口这样从外部给电路板上的元器件供电的都属于电源，通常我们可以在电路板或者原理图上找到以下标识：

V_{IN}

连接在电路板上的外部电压源，很多电路板支持一系列的输入电压，并在内部提供一个内部电压控制以在休眠状态时提供给其他元器件供电。

V_{CC} 或 V_{DD}

电路板内部的电压控制器，它们一般提供的电压为+5v，+3.3v，+1.8。

GND

地，电路板上的 0v 参考点，板子上的电压都是根据 GND 来衡量的。

3.模拟和数字 I/O

将外围设备的不同种类的 I/O 连接到你的开发板上 GPIO 上，输入引脚允许你的应用读取和理解该引脚当前的电气状态，输出引脚则允许你控制该引脚的电气状态，外置设备和板子上的 GPIO 都有数字和模拟两种。

模拟 I/O 的设备产生的电压值是和传感器采集到的物理值成比例的，举个例子，一个温度传感器模块，可能能在信号输出脚产生与 0-100°C 对应的 0-5v 的电压值。

在实际电路中，通常使用一个模数转换器(ADC)来将模拟的电压值转换为一个数字的数据，ADC 的分辨率是用来表示 ADC 转换后的整数范围，例如，一个 10 位的 ADC，能表示的范围就是 0-1023(2^{10})。数字 I/O。

数字电路将电路的状态表示为二进制的值：

高电平：当电压达到或接近电源电压的时候，通常表示为“1”。

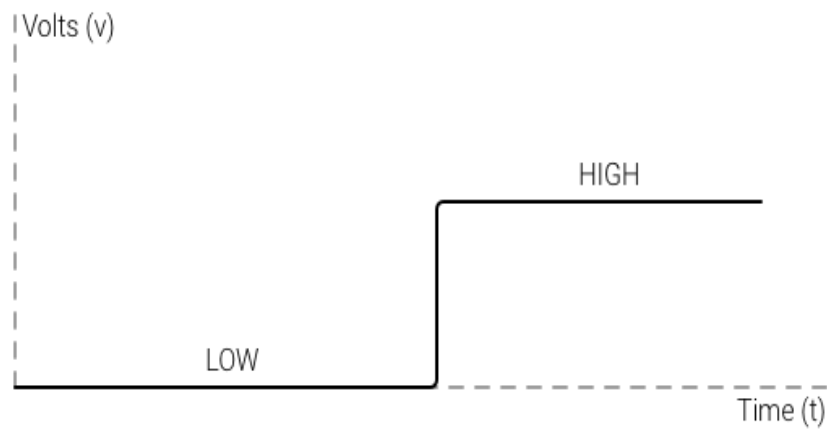
低电平：当电压达到或接近地的电压的时候，通常表示为“0”。

很少有某个数字信号完全达到 0V 或电源电压，大多数数字逻辑设备取一个电压范围判断为一个有效的逻辑电平。下表显示为每个逻辑状态常见的输入电压范围：

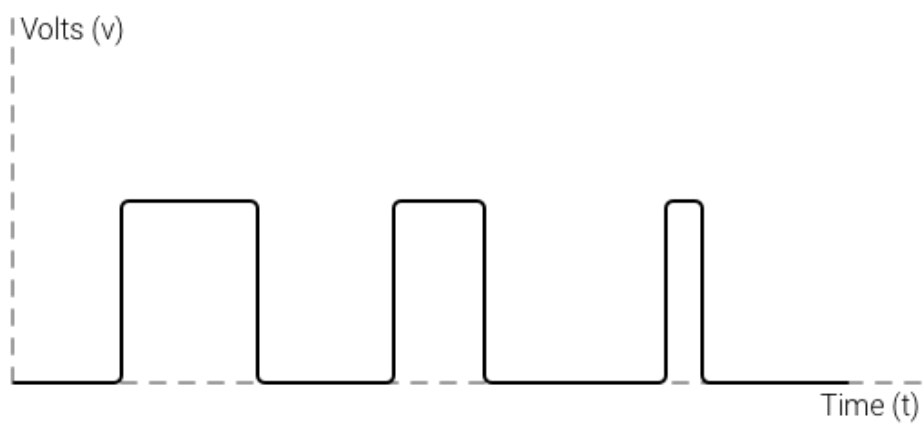
Supply Voltage (VCC)	Logic Low (0)	Logic High (1)
5V (TTL)	< 0.8V	> 2.0V
3.3V (CMOS)	< 0.8V	> 2.0V
1.8V (CMOS)	< 0.6V	> 1.2V

对于数字 I/O，通常有下列几种使用情况：

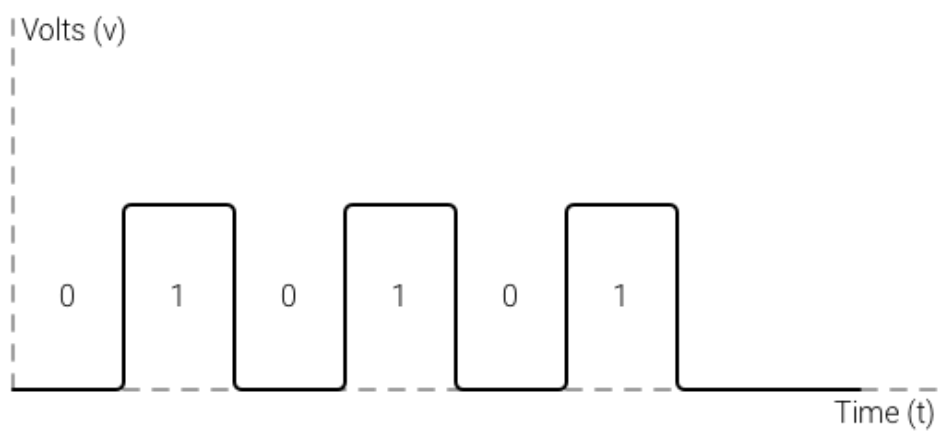
稳定状态：端口表现出一个稳定的单一状态，高电平或低电平



脉冲序列:一系列数字脉冲变频信号和宽度随着时间的推移不断传播。



串行通信:一系列数字 1 和 0 代表各个 bit 的二进制数

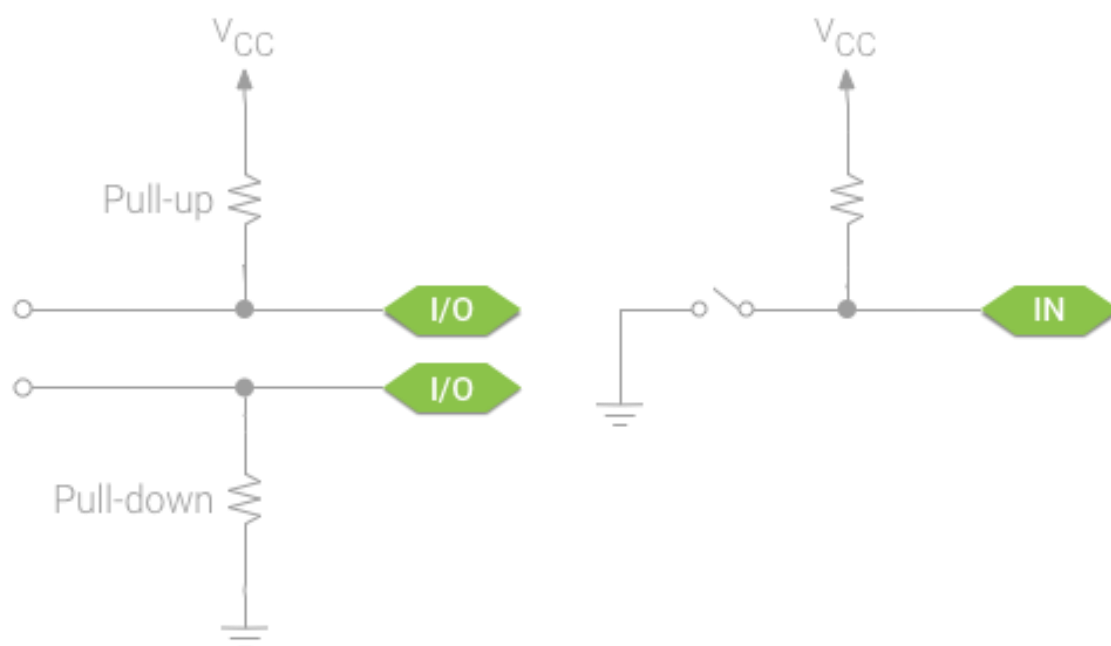


更多关于 I/O 的信息，可以查阅数字电路和单片机的相关知识。

4.上拉和下拉

在许多数字接口电路中,I/O 引脚通过电阻连接和 VCC 或地面。这些电阻分别被称为上拉和下拉电阻。他们保证每个引脚都有一个稳定的默认状态,系统的其余部分可以正常工作,没有显著影响直接输入或输出信号。

数字信号输入端口不能很好的读取悬空的引脚,悬空的引脚容易受到电磁干扰,从而使你获得一个不正确的读数,上拉或下拉电阻确保信号线趋于一个稳定值。



举个例子,有一个按钮或开关。当一个开关闭合时,引脚连接低电平或高电平,但开关松开时引脚悬空。此外,许多数字传感器使用集电极开路(或开漏)输出报告状态改变。当开关是开着的时候,这些输出像一个简单的开关,需要外部源来驱动输入。

电阻强度

不同的电阻的值以不同的方式影响系统。低阻值电阻被认为是“强”,因为可以通过更多的电流。强大的上拉(或在下拉)吸引更多的能量,但他们使信号闲置的水平(即信号失效)比一个“弱”高阻值电阻器更快。

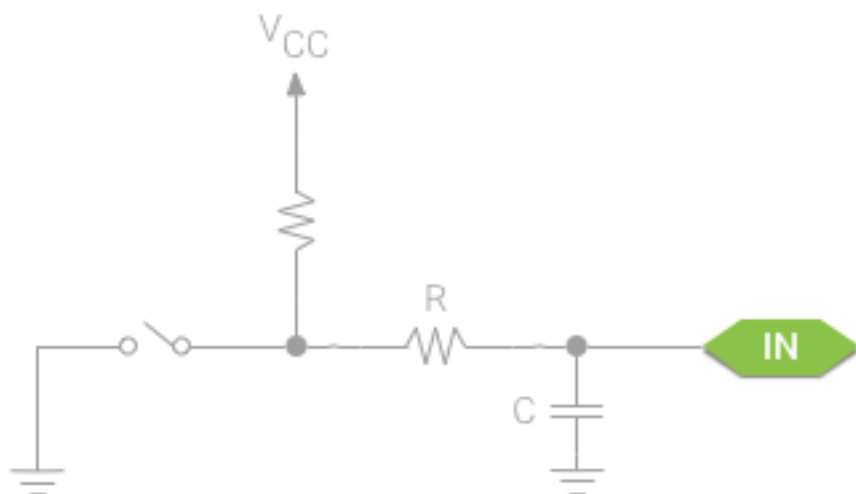
上下拉的电阻阻值通常为 1k-10k 之间。

5.信号消抖

许多电子输入设备,如开关和继电器、机械组件。在器件变换状态的那一瞬间,电信号可以暂时振荡或在多个值之间的“反弹”。比如你按一下按钮,按钮并不是立刻开或关,而是要经过数毫秒的“抖动”,这个时间非常短,但是会造成你的应用读取到多次输入。

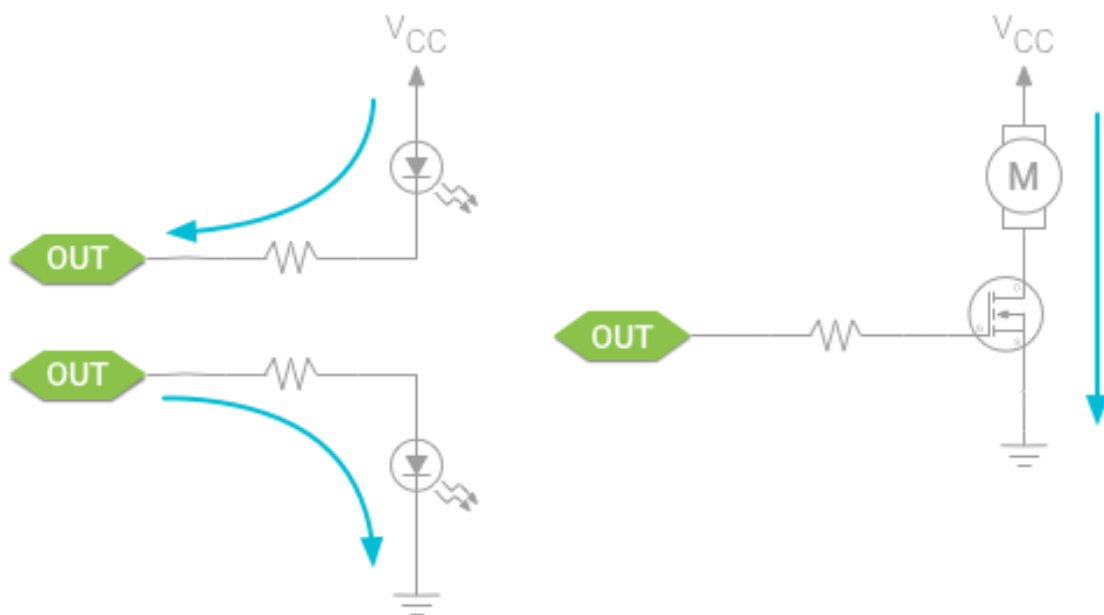
要解决这个问题,你必须使用硬件或软件防抖信号。软件防抖包括设置一个初始输入事件之间的时间延迟(通常不超过几百毫秒),输入预计将稳定。

硬件防抖,添加一个简单 RC 电路(它包含一个电阻器和电容器)和输入引脚之间的设备。当输入设备的状态发生变化时,电容器充电和放电速度将与输入电阻的大小成比例,有效地减缓信号跳变。



6.保护 I/O 引脚

每个输出端口具有有限的承受外部拉电流(高电平)或灌电流(低电平)的能力。外围设备会吸引比端口可以处理的更多的电流，甚至会损害输出端口。为了保护端口，可以在负载和端口之间串联一个限流电阻。



通常限流电阻阻值在 100Ω - $1K\Omega$ 之间。

控制高功率转换器,如电机、输出端口到负载的缓冲，直接从电源转换器使用晶体管或类似的电子控制开关和电源。

不同设备可以承受的拉电流和灌电流是不一样的，仔细阅读你的硬件手册可以获得更多信息。

为了安全操作，所有的端口请置于安全的电压范围(0V-VCC)，任何高于 VCC 的电压都有可能损坏开发板，请多次确认传感器或转换器的电压范围和开发板的电压范围是匹配的，如果确实不同，请使用电平转换电路。

四、套件各个模块介绍

本套件包含的模块主要有：

面包板	*1
10 厘米 40p 双公头杜邦线	*1
20 厘米 40p 公对母杜邦线	*1
蜂鸣器模块	*1
MC-38 门磁开关	*1
金属膜电阻	*30
发光二极管	*3
轻触开关带键帽	*3

1.面包板

本文第二章节已经介绍过面包板了，面包板上有许多内部连接在一起的孔洞，可以将元器件插在上面做一些简单的实验。

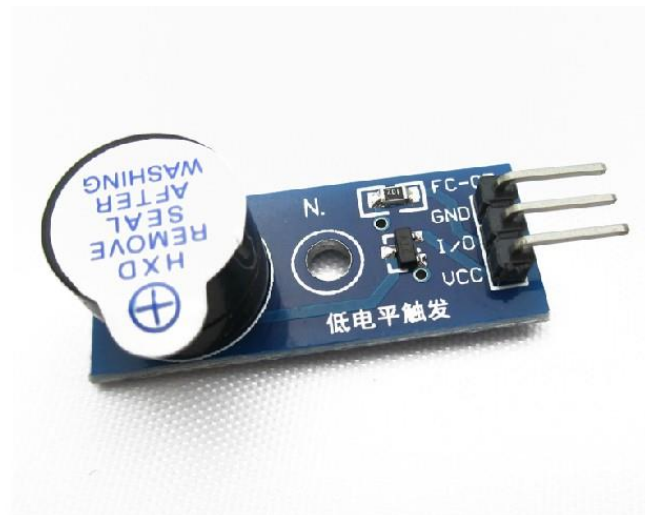


2.杜邦线



用于连接开发板，面包板和模块的导线，有公头和母头之分，可以很方便的插入和拔出。

3.蜂鸣器模块



本套件的蜂鸣器采用有源蜂鸣器，低电平触发，即在模块信号输入端给一个低电平，蜂鸣器就可以鸣叫。

4.电阻器



限流元件，将电阻接在电路中后，电阻器的阻值是固定的一般是两个引脚，它可限制通过它所连支路的电流大小。

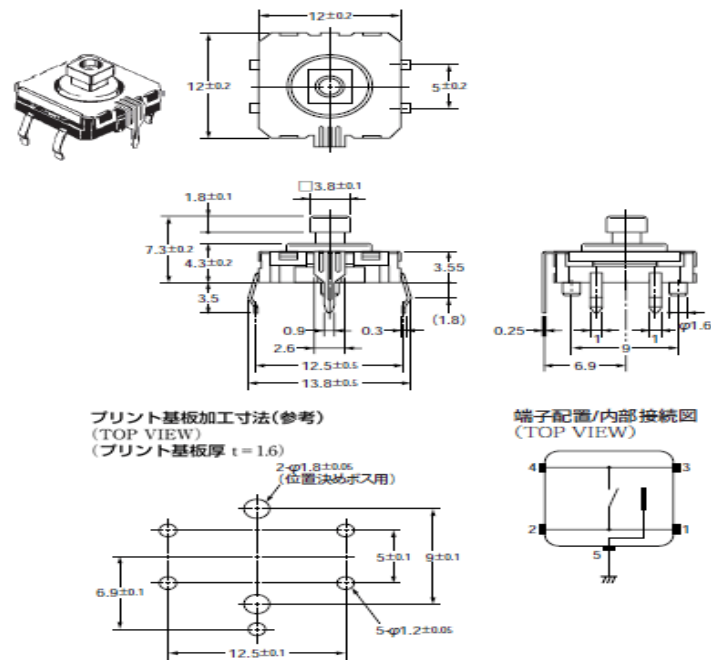
5.发光二极管



LED 有两根引脚，长正短负，本套件采用的 LED 有红黄绿三种颜色，各个颜色的数量不定，但总数一共 9 个。

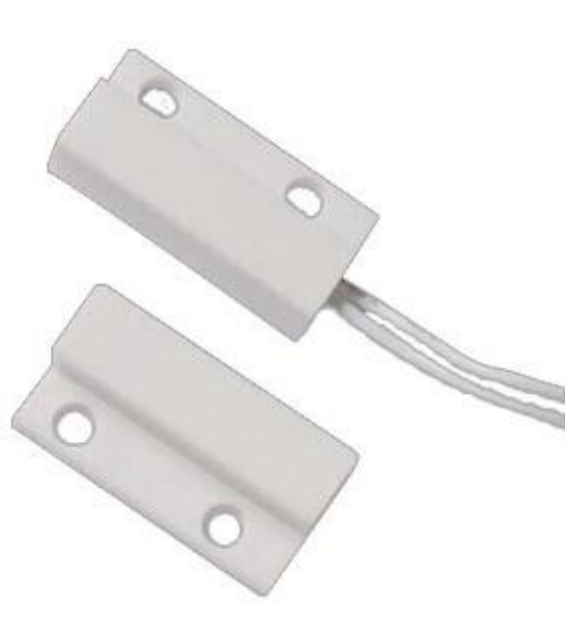
6.轻触开关

套件包含 9 个外形尺寸为 12mm*12mm*7.3mm 的轻触开关，
同时按键会附带 9 个按键帽



如上图所示开关内部接线，按下按键的时候 1 脚，2 脚会和
3 脚，4 脚联通。

7.门磁开关



动作距离：18mm \pm 6mm

开关耐压：110VDC

负载电流：500mA

寿命：100 万次

开关输出：常闭(门磁开关常闭是指:开关与磁铁靠在一起时开关是闭合的,开关与磁铁分开时开关是断开的.)

明装式，适用于非铁质门或窗 表面安装

五、注册 OneNET 平台

OneNET 接入过程大致如下：



用户在接入 OneNET 之前，必须先在 OneNET 平台注册用户账户，用户注册登录成功后，可以在用户账户下创建产品，目前平台提供有公开协议产品和私有协议产品两种产品类型（平台为公开协议产品又提供有 HTTP、EDP、MQTT、MODBUS 等多种协议的选择）。在产品中创建设备，为设备新增数据流。设备端编写终端接入代码，主要完成数据采集、协议封装、数据上传等工作，终端设备的数据上传成功后，平台在相应数据流下会生成随时间推移的数据点。最后，为了更直观的呈现数据的变化情况，用户可以运用应用孵化器自定义个性化应用并发布。

用户注册：

为了使用 OneNET 设备云的强大功能，您首要做的是在 OneNET 上注册您的开发者账号，来创建您专属的“开发者中心”；

登陆 <http://open.iot.10086.cn/>

点击首页右上角的“注册”按钮，注册用户账号；

填写用户名、用户密码、有效邮箱地址(或者有效手机号码)等，点击获取验证码，打开邮箱邮件查看验证码，并完成注册；

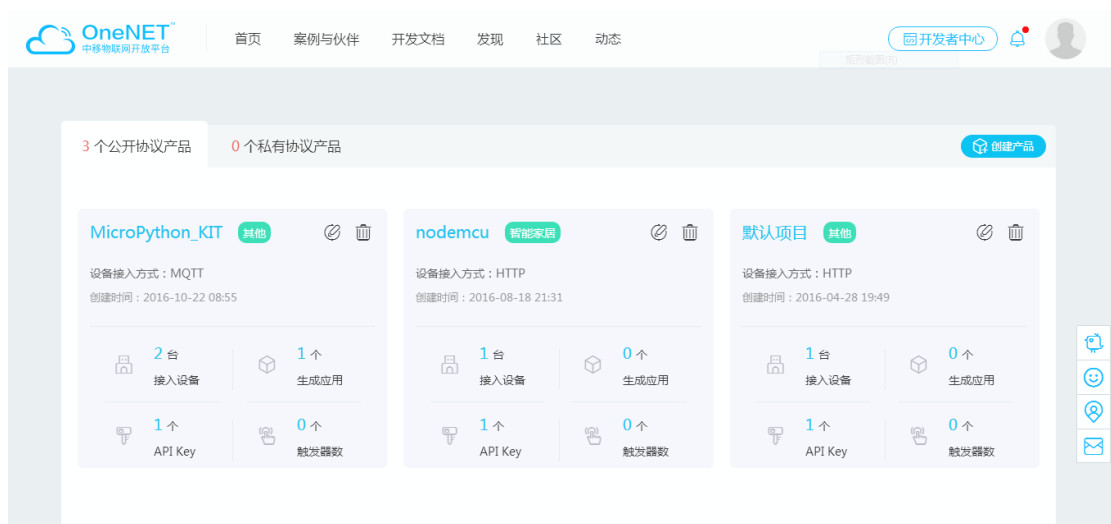
OneNET 支持“个人用户”和“企业用户”两种入驻方式，您可以根据您的实际情况选择注册方式；



注册完成后，回到主页点击“登录”，即可进入 OneNET 的官方主页，并由此进入到您的“开发者中心”。

接下来，为了使用 OneNET 的强大功能和服务，您需要在 OneNET 平台上创建您的产品；

点击“开发者中心”，进入相应的“产品列表”管理页面，在这里您可以新建并管理您的产品；



点击右上角的“创建产品”，在弹出页面中按照提示填写产品的基本信息，进行产品创建；

在创建过程最后一步，系统会提示让您选择“设备接入方式”和“设备接入协议”，OneNET 提供公开协议和私有协议两种产品类型。

创建完成后，在弹出的对话框可进一步创建设备或者返回到产品列表页面；

在产品列表页面会列出您创建的全部产品，可直接对各产品进行编辑和删除操作，也可以点击产品名称，进入该产品的管理页面；



点击设备管理，然后点击添加设备按钮，在填入设备名称，鉴权信息后，就完成设备的添加了，添加设备后我们需要记住三个参数用于设备的登陆：

创建设备时得到的设备 ID，为数字字符串；

注册产品时，平台分配的产品 ID，为数字字符串；

设备的鉴权信息（即唯一设备编号，SN），或者为 apiKey，为字符串。

更多关于开发者文档的内容，可以参考：

<http://open.iot.10086.cn/doc>

查看设备 ID:

设备数量: 2个 设备接入协议: MQTT

输入设备ID或者设备名称

全部 全部



在线

MQTT测试设备

设备ID:3264518

创建时间:2016-08-17 14:34:52

查看产品 ID:

OneNET

产品概况

设备管理

其它

MicroPython_KIT

芝麻DIY micropython for onenet 套件

产品ID : 74321 | 设备接入协议 : MQTT |

用户ID (user id) : 29477

设置鉴权信息：

接入设备 MQTT协议

• 设备名称:

设备名称

• 鉴权信息:

请输入自定义字符串

设备间不能设置相同的字符串，最多64个字母、数字或字母与数字组合的字符串。

• 数据保密性:



私有



公开

接入设备

取消

六、例程解析

完成所有准备工作后，我们就可以将设备接入到平台上了，本节通过六个实例，介绍 MicroPython for ONENET 入门套件的一些使用方法，包括平台控制开发板上的 LED，上传 DHT11 数据到平台，设备间通信等一些基本的用法，大家可根据这些修改，重写，用自己的方法实现自己想要的效果。

1. 平台控制 LED

我们使用搭载 ESP8266 模块的 MicroPython 开发板，使用了 umqtt.simple.py 模块，该模块是对 MQTT 协议的封装，我们使用 MQTT 协议，接入中国移动 ONENET 物联网平台，我们的目的是在物联网平台上发送命令，控制开发板上自带的 LED，我们发送 ‘on’，使得 LED 亮起，发送 ‘off’ 使得 LED 熄灭。

首先我们打开例程文件夹，可以看到 boot.py 和 main.py 两个文件，其中 boot.py 是启动时运行，用来连接本地的 WIFI，并且在连接 WIFI 后输出一些网络信息。boot.py 程序第 10 行的代码是本地的 WIFI 的 SSID 和密码，大家修改成自己的就可以连接上自己的 WIFI 了。如我的 wifi 名为:MERCURY_D5C8 密码为 zhimadiy 则：

```
sta_if.connect('MERCURY_D5C8', 'zhimadiy') #wifi 的 SSID 和密码
```

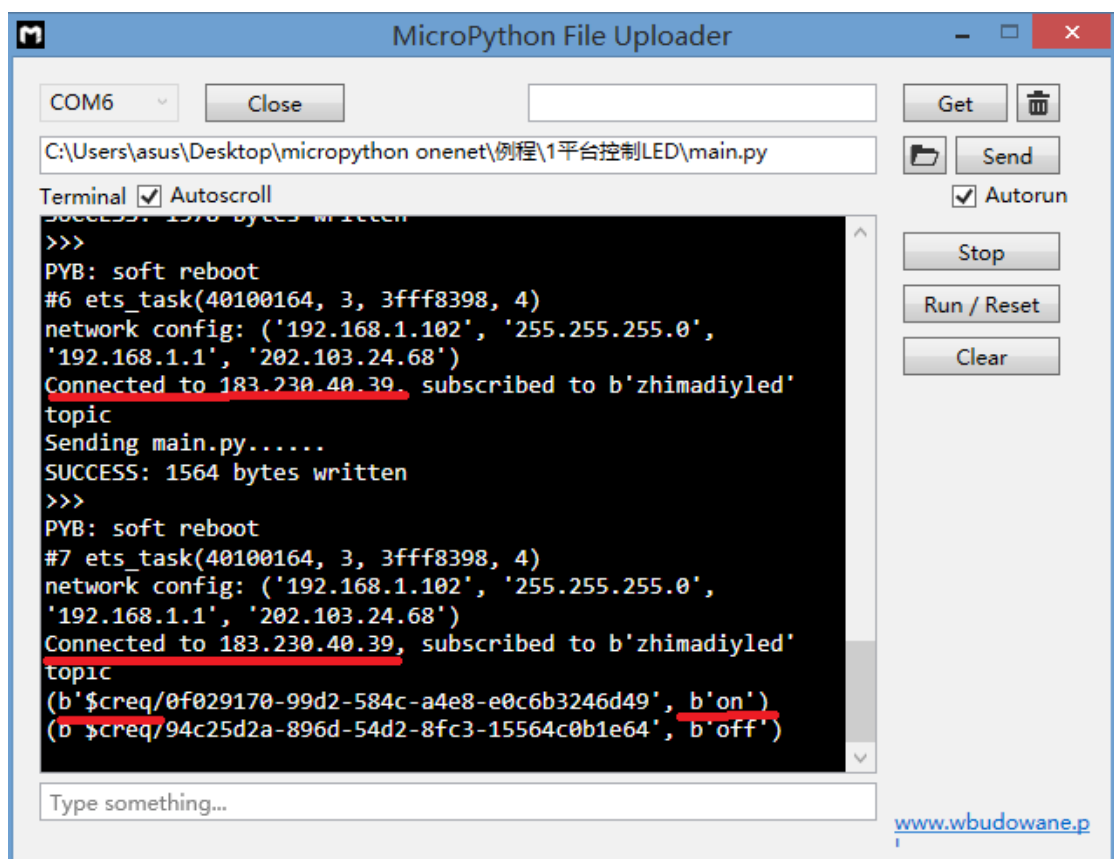
修改好 boot.py 后，继续修改 main.py, main.py 是主程序，在 boot.py 运行完后运行，我们打开 main.py 只需要修改 CLIENT_ID, username, password 三个参数为自己的就可以了

```
7  #MQTT服务端信息，使用时修改成自己的
8  SERVER = "183.230.40.39"    #ONENET服务器地址
9  SERVER_PORT=6002           #ONENET服务器端口
10 CLIENT_ID = "3784858"      #创建设备时得到的设备ID，为数字字符串
11 TOPIC = b"$dp"              #ONENET上传数据点需要传到此TOPIC
12 username='74321'           #注册产品时，平台分配的产品ID，为数字字符串
13 password='zhimadiymicropythonesp8266' #鉴权信息
14 message="{\"hello\":1}"      #即{"hello":1},向名为hello的数据流传1
```

修改完成后，使用工具上传到板子上，我们使用的工具为 MicroPython File Uploader.exe，大家可以到资料内的软件工具文件夹获取到，也可以到网上下载得到。

此工具必须在连接好开发板到电脑以后才能打开，连接好以后，我们首先点击 open 按钮打开相应串口，然后点击文件夹图标选择需要上传的文件，最后点击 Send 按钮发送文件，发送完成后，板子会自动重启。

上传到板子上以后，我们就可以看到如下界面：



当看到 Connected to 183.230.40.39 时，说明已经连接成功，我们就可以看到网页上绿灯亮起



此时点击图中红圈处图标，发送” on”，可以使板子上的 LED 亮起，发送 off，可以使 LED 熄灭。

MQTT协议下发命令

• 命令内容:

on

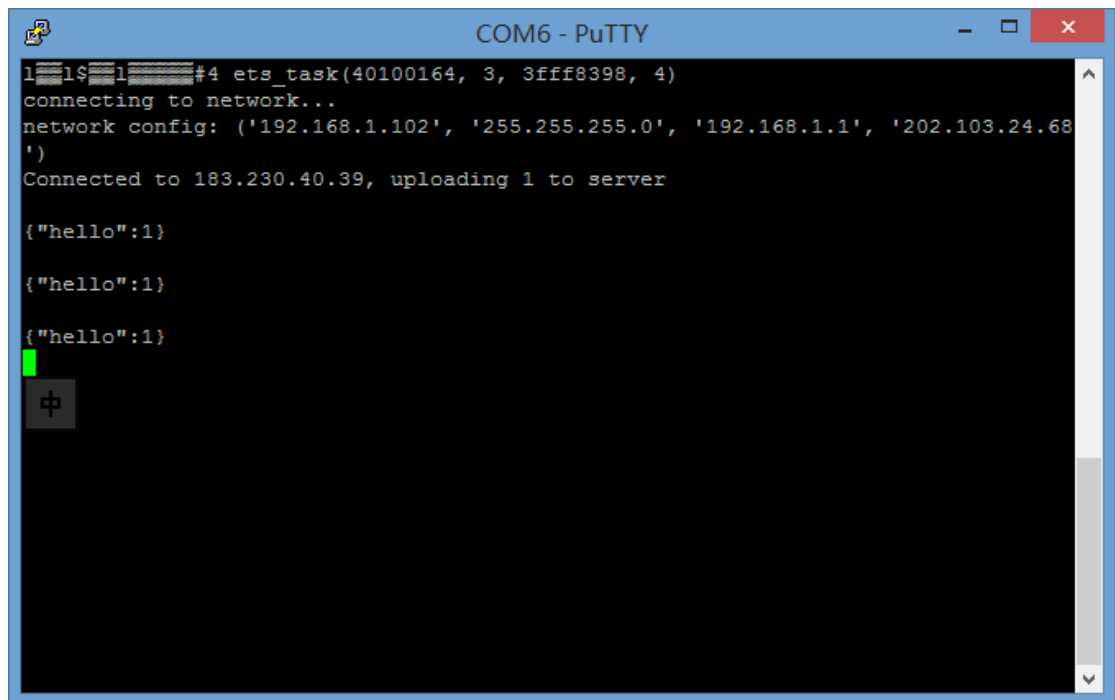
发送命令

取消

2. 向平台上传静态数据

我们首先按照例程一的方法修改 boot.py 和 main.py,

修改好以后上传到开发板上, 如果一切正常, 板子每隔几秒就会向平台发送一串固定的数据, 我们可以在开发板的串口上观察到一下输出信息:



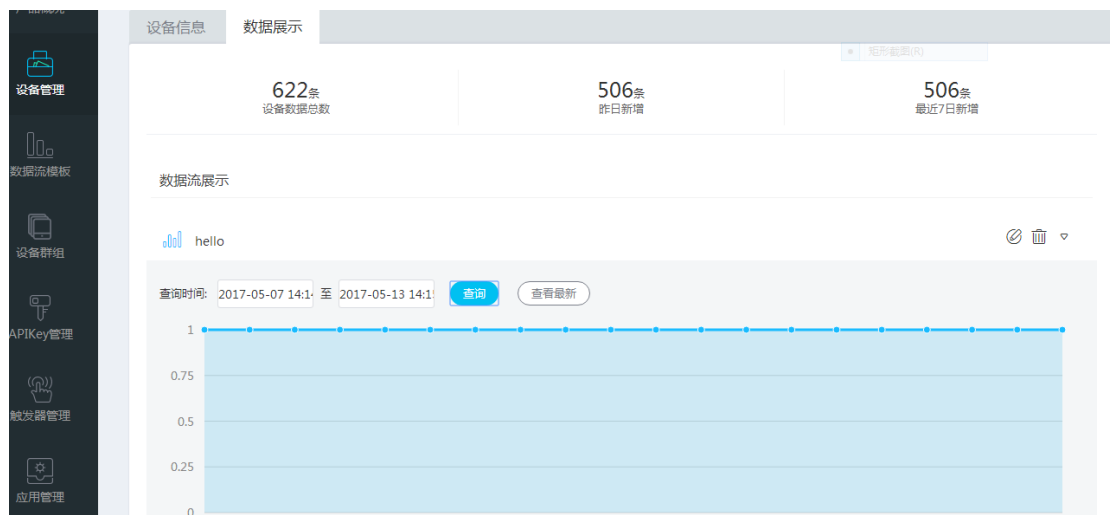
```
l1$ ./l1.py #4 ets_task(40100164, 3, 3fff8398, 4)
connecting to network...
network config: ('192.168.1.102', '255.255.255.0', '192.168.1.1', '202.103.24.68')
Connected to 183.230.40.39, uploading 1 to server

{"hello":1}

{"hello":1}

{"hello":1}
```

此时在平台的网页上我们可以观察到上传的信息:



这说明, 开发板上传数据成功。

在这个例程的 main.py 文件中我们可以看到代码的第 14 行：

```
message="{\"hello\":1}"      #即{"hello":1},向名为 hello 的数据流传 1
```

这一行代码创建了一个字符串，用于上传到平台，即{"hello":1}，向名为 hello 的数据流传 1。

同时我们也可以看到：

```
TOPIC = b"$dp"              #ONENET 上传数据点需要传到此 TOPIC
```

OneNET 要求上传数据点必须上传到"\$dp"

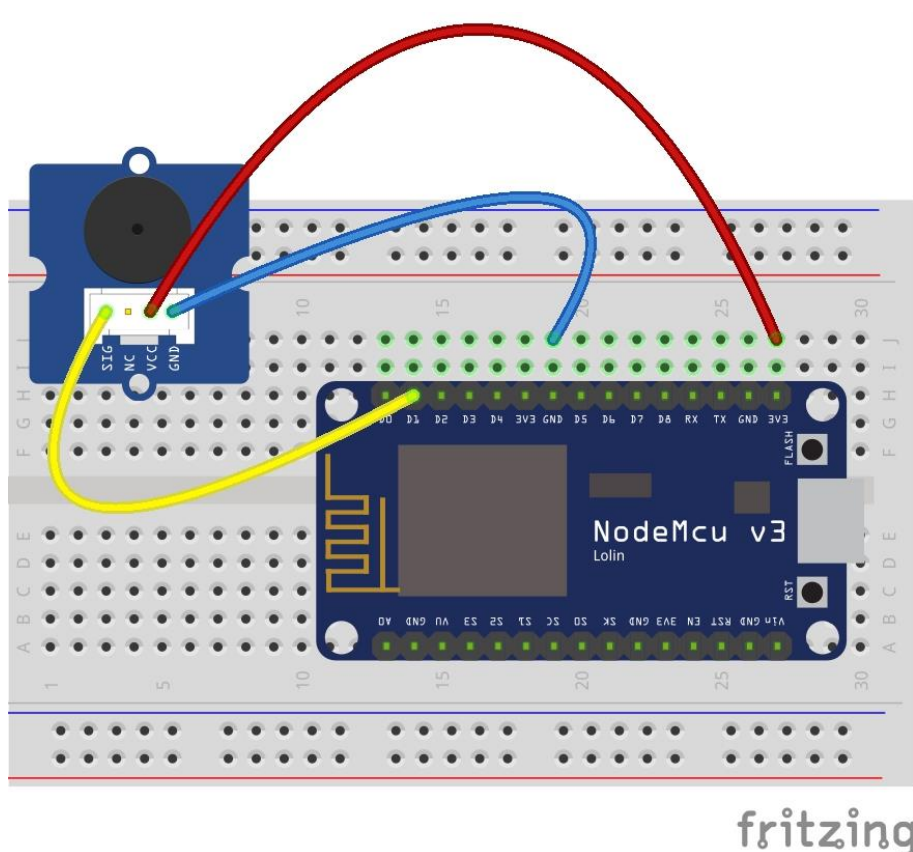
在代码的后半段我们可以看到：

```
16 msglen=len(message)
17 tmp=[0,0,0]
18 tmp[0]='\x03'
19 tmp[1]=msglen>>8
20 tmp[2]=msglen&0xFF
21 message="%c%c%c%s"%(tmp[0],tmp[1],tmp[2],message)    #将消息封装为ONENET要求的格式
22
23 def main(server=SERVER):
24     c = MQTTClient(CLIENT_ID, server,SERVER_PORT,username,password)
25     c.connect()
26     print("Connected to %s, uploading 1 to server" % server)
27     while True:
28         while True:
29             time.sleep_ms(2000)
30             print(message)
31             c.publish(TOPIC, message)
32             time.sleep_ms(2000)
33
34     c.disconnect()
35
36 main()
```

此部分代码表示，将需要上传的字符串打包后，在 main() 函数内连接服务器，每隔 4 秒循环向服务器的"\$dp" 上传字符串。

3. 控制蜂鸣器

这一个例程的源码基本和例程一差不多，都是使用平台下发命令控制开发板，我们要先按例程一的方法修改好代码，但在上传代码之前，我们要按图下的方法连接好电路：

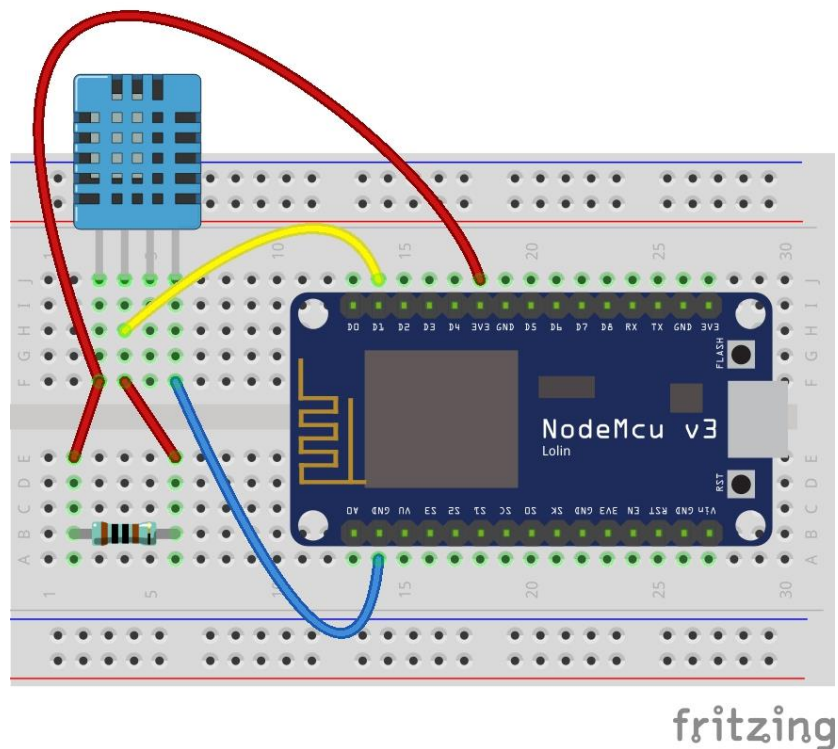


套件中的蜂鸣器模块和图中的有些差别，但使用上都是一样的，开发板 D1 连接模块信号端，VCC 接 VCC，GND 接 GND，连接好以后，将代码上传到开发板上，如果一切正常，从平台 MQTT 命令下发“on”就可以使蜂鸣器响起，发送“off”停止响声。

我们在线路图文件夹找到外置 LED. jpg 文件，蜂鸣器程序无需修改即可工作于该电路。

4. 上传 DHT11 温湿度数据

我们还是按照例程 1 的方法修改好源码，上传到板子上，然后按照下图连接好电路：



当板子正确运行程序时我们可以观察到：

```
COM6 Close [ ] Get [ ]
C:\Users\asus\Desktop\micropython onenet\例程\4上传DHT11温湿度数据
Terminal [x] Autoscroll
temperature is 28,humidity is 46
{"temperature":28,"humidity":46}
temperature is 28,humidity is 46
Sending main.py.....
SUCCESS: 1612 bytes written
>>>
PYB: soft reboot
#7 ets_task(40100164, 3, 3fff8398, 4)
network config: ('192.168.1.102', '255.255.255.0',
'192.168.1.1', '202.103.24.68')
Connected to 183.230.40.39, uploading DHT11 data to server
temperature is 28,humidity is 46
{"temperature":28,"humidity":46}
temperature is 28,humidity is 46
{"temperature":28,"humidity":46}
temperature is 28,humidity is 46
{"temperature":28,"humidity":46}
temperature is 28,humidity is 46
{"temperature":28,"humidity":46}
Type something... [ ] www.wbudowane.p
```


开发板不断向平台上传温湿度信息。

我们查看源码可以看到：

```
from dht import DHT11
```

导入 DHT11 库，使得读取温湿度更方便

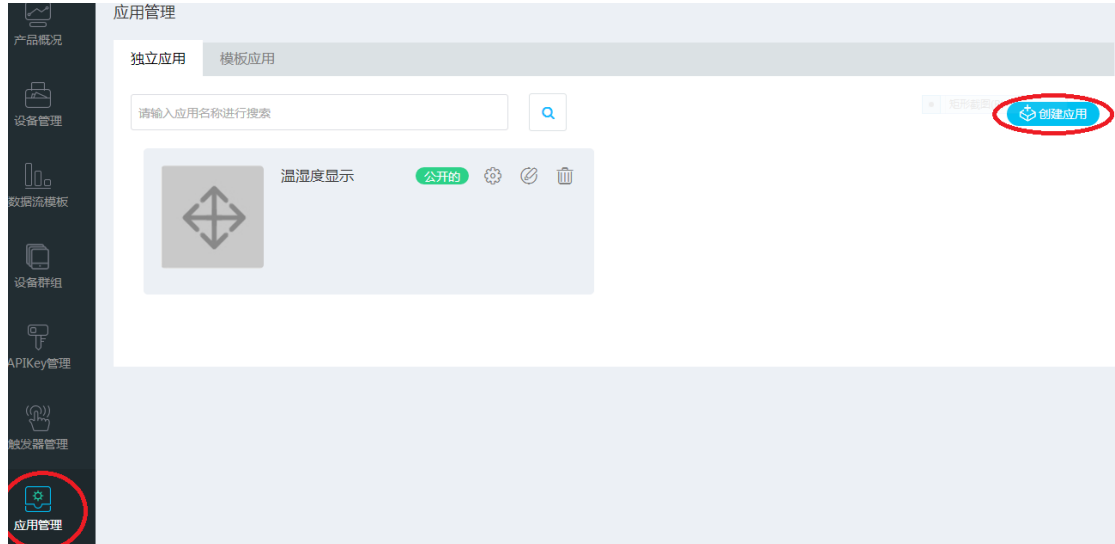
```
def pack_msg(message)
```

定义消息封装函数，要发送的消息必须打包为 OneNET 要求的格式

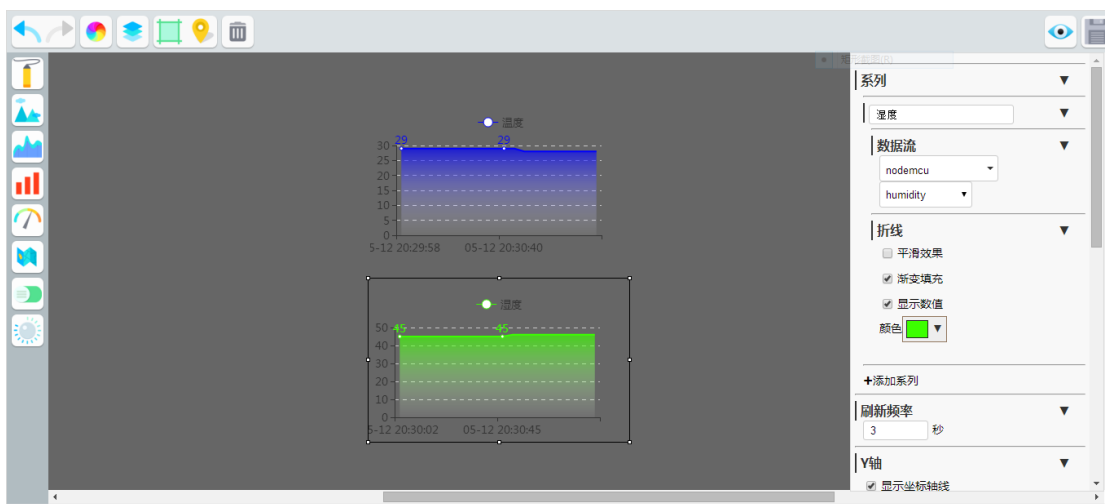
```
msg="{\"temperature\":%d,\"humidity\":%d}\"%(tem,hum)
msg=pack_msg(msg)
print(msg)
c.publish(TOPIC, msg)
```

同时上传温湿度两个数据。

当平台可以接收到来自开发板的消息后，为了更直观的呈现数据的变化情况，我们可以运用应用孵化器自定义个性化应用并发布。



点击应用管理下的创建应用按钮即可创建应用，在应用创建器界面中，我们点击左侧样式栏中的折线图，分别创建名为温度和湿度的折线图，选择合适的数据流，即可看到实时上传的数据点。

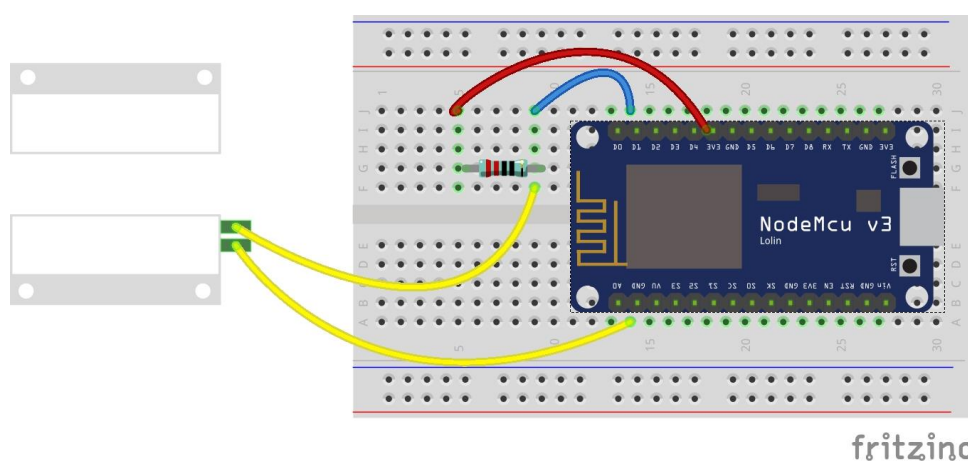


点击右上角保存按钮即可发布应用。

5. 门磁开关

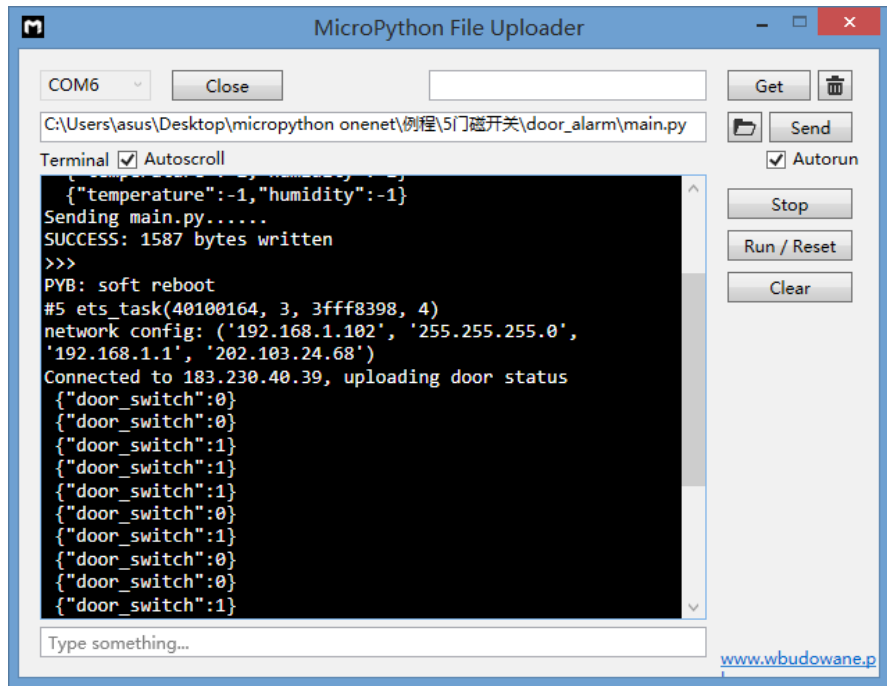
随着物联网进程的不断加速，许多传统的设备被接入网络，在以前，防盗报警器主要通过电话网络达到报警效果，比如主人不在家的时候，在家里设置好报警设备，当传感器检测到人员走动或门被开关，就会向主人手机拨打电话或发送短信，这种报警设备主要由一个报警主机和多个检测器组成，成本较高，随着开源硬件的普及，我们可以自己动手打造一款这样的设备。

我们首先按照例程 1 修改好源码，然后上传的开发板上，然后按照下图的方式连接好电路图：



我们本次选用的 MC-38 门磁开关是常闭型，即当两端闭合时导通，从 fritzing 的电路图中我们可以看到，D1 通过一个 1K 的电阻接到 VCC 端，而 GND 通过门磁开关接到 D1 端，这样，当门磁开关断开即门被打开时，D1 端被上拉处于高电平状态，当门被关闭时，D1 端接地，处于低电平状态。

上传完成后可以观察到：



当修改上传完程序时，已经可以向平台发送门磁开关的状态了，

但是如果要达到报警效果，我们需要到平台添加触发器：

设备管理

数据流模板

设备群组

APIKey管理

触发器管理

应用管理

第三方开发平台

触发器信息

• 触发器名称：门磁报警

• 控制范围：Micropython_ZM ☐ 全部设备

• 数据流名称：door_switch

• 触发条件：选中数据流值 change 数据值

接受信息方式

☒ 邮箱 ☐ URL

邮箱：

提示：每天最多发送20封邮件

添加触发器 取消

设置好设备名，数据流名以后，我们设置为当数据改变时发送邮件通知，这样当门磁开关的状态改变时，我们就可以收到邮件通知了：



我们可以看到，已经收到 OneNET 的邮件通知了。

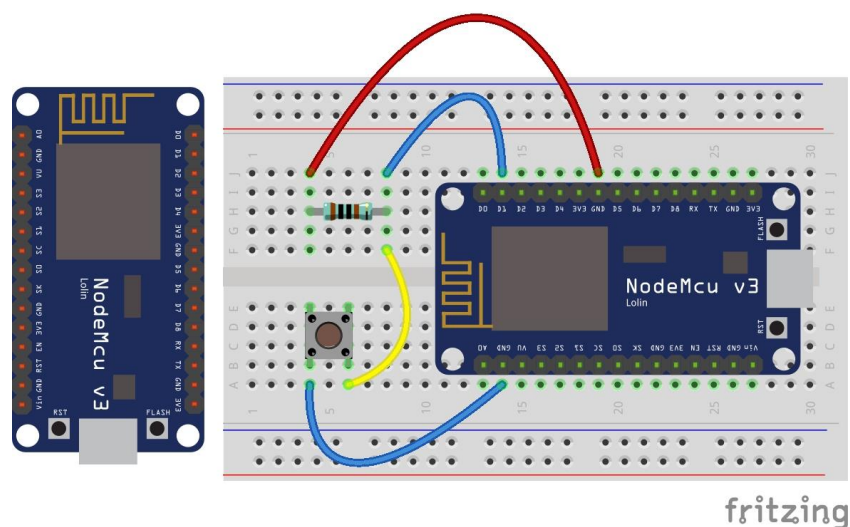
6. 设备间通信

基于 Topic 的订阅、发布以及消息推送，可以实现设备间的消息单播以及组播，MQTT 协议十分适合嵌入式设备的组网，本节介绍一种简单的组网方法，使用一块开发板，控制另一块板子上的 LED。

其中控制方板子使用外部按钮，每当按下的按钮，向服务端发送的消息就由{“notice” :1}变为{“notice” :0}：

```
while True:
    if door_switch.value()==0:
        msg="{\"notice\":0}"
        led.value(1)
    else:
        msg="{\"notice\":1}"
        led.value(0)
```

我们按照例程 1 的方法修改上传完程序到板子以后，按照下图连接好硬件：



例程文件夹中的 `button` 程序上传到右边的开发板上，
`led_on_board` 程序上传到左边的板子上，这样当双方都运行的时候，
一块板子就会监听来自控制方板子发出的数据流，控制方通过按钮即可控制左侧板子的 LED。

需要注意的是此时监听方的源码.

例程/led_on_board/main.py:

```
TOPIC = b"/6046214/notice"
```

通过订阅 /device_id/数据流名 的方式, 及时获取到某设备最新的数据点信息。