MICO Documentation Working Group (MDWG) Jenny Liu
Track Number: 0004CN MXCHIP Co., Ltd
Version: 1.0 June 2015
Category: Technical Manual

MiCOKit 网络配置协议

摘要（Abstract）

本文档主要介绍用户 APP 给 MiCOKit 设备配置 Wi-Fi 网络连接、设置设备参数 以及配置设备接入 FogCloud 云的方法。使用户对 MicoKit 网络配置有一个初步的认识，方便 MiCO-IoT 物联网开发者使用 MiCOKit 开发板及 FogCloud 云进行物联网应用开发。

适用对象（Suitable Readers）

本文适于 MiCO 智能设备开发者学习参考，尤其适用于 APP 端开发人员。

获取更多帮助（More Help）

MiCO 开发团队向您推荐：MiCO 开发者学习网站：http://mico.io/（至开发者中心），获取更多最新资料。

手机微信“扫一扫”关注：“MiCO 总动员”公众号，获取 MiCO 团队小伙伴最新活动信息。

登录上海庆科官方网站：http://mxchip.com/，获取公司最新产品信息。

Table of Contents

List of Figures

List of Tables

1. 概述

本文主要介绍用户 APP 给 MiCOKit 设备配置 Wi-Fi 网络连接、设置设备参数，以及配置设备接入 FogCloud 云的方法。

MiCOKit 设备工作模式有以下两种。

1.1. 配置模式

该模式主要实现用户设备 Wi-Fi 网络配置和设备参数设置。

（1）设备收到 APP 发送的 SSID/key 并成功连接路由器后，就会自动连接用户 APP 的 FTC Server，并发送设备的当前配置信息给 APP。

（2）此时用户 APP 可以修改配置参数，并发送给设备，完成配置。具体通信方式参照 Easylink APP 的实现(IOS/Android)。

1.2. 正常工作模式

设备 Wi-Fi 网络和设备参数配置完成后，自动重启设备，进入正常工作模式。

此时设备默认开启 Congifg Server(TCP Server)，用户 APP 作为 TCP client 与之连接。

APP 使用 mDNS 协议发现设备，并与之建立 TCP 连接，之后通过 HTTP 协议与设备进行数据交互，数据包采用 JSON 格式。

2. 设备 Wi-Fi 配置

MiCOKit 设备支持 Easylink 配网模式，用户 APP 使用 Easylink 协议给设备配置 Wi-Fi 网络。

配置方法：

（1）短按设备上的 Easylink 按键，使设备进入 Easylink 配网模式（绿灯快闪）；

（2）打开 APP 上的 Easylink 配网功能，输入要连接的 Wi-Fi 密码，开始配置；

（3）设备收到 Wi-Fi 的 SSID 和密码后（绿灯慢闪）自动连接 AP， 连接成功后（绿灯常亮）自动重启，进入正常工作模式，Wi-Fi 配置完成（绿灯常亮）。

3. 设备参数设置

在使用 Easylink 协议配置设备 Wi-Fi 的过程中，还可以设置设备的参数。

设备收到 APP 发送的 SSID 和密码并成功连接 AP 后，会将设备的当前配置参数发送给 APP，用户在 APP 上可修改这些参数并写入设备。

以上二、三步骤设备均工作在配置模式，详细实现参考 Easylink 协议以及 Easylink demo APP 的实现。

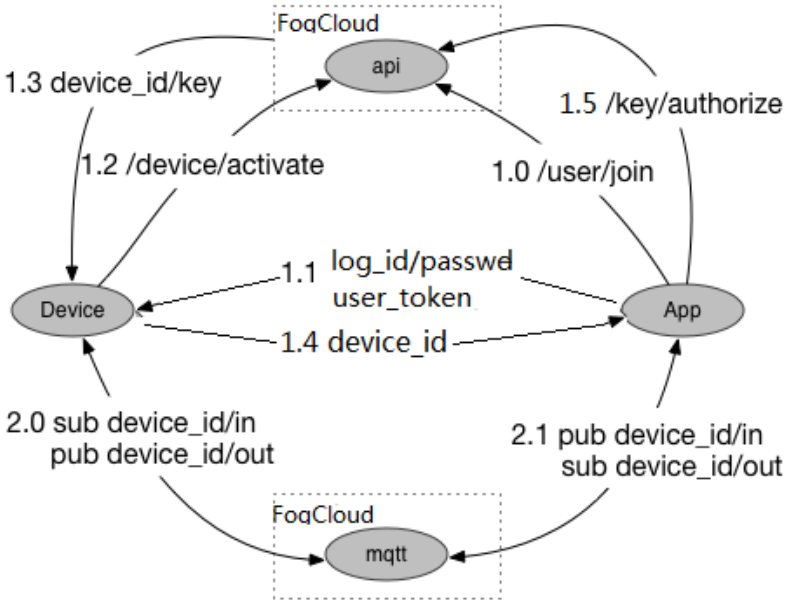
4. 设备云连接配置

设备配置完成后，重启进入正常工作模式；设备 Wi-Fi 连接成功后，开启 FogCloud Congifg Server(TCP Server)，用户 APP 作为 TCP client 可与之建立连接。APP 使用 mDNS 协议发现设备，并与之建立 TCP 连接，之后通过 HTTP 协议与设备进行数据交互，完成设备的激活、授权、重置等请求，使得设备接入 FogCloud 云端并和用户 APP 绑定。

通信数据包采用 JSON 格式，具体通信方式如下：

4.1. 设备接入 FogCloud 云流程

Figure 1. 设备接入 FogCloud 流程



（1）设备首次接入 FogCloud 云之前，需要先激活设备；APP 向设备发送激活请求，使得设备向 FogCloud 云端激活，成功后返回设备 ID 给 APP，完成设备激活和绑定；如上图 1 中的 1.1，1.2，1.3，1.4，1.5；

（2）设备激活成功后，重启后自动连接 FogCloud 云消息服务器（上图 1 中 2.0）；

（3）之后其他 APP 要绑定设备，只需要向设备发送授权请求（如果发送激活请求，则设备实际执行授权），设备向云端请求 APP 授权，返回设备 ID 给 APP（过程同步骤(1)）；

（4）激活（或授权）完成后，APP 通过设备 ID 向 FogCloud 云端查询设备连接状态，并可与设备间进行云端消息收发（上图 1 中的 2.1）。

4.2. APP 请求接口

4.2.1 查询设备状态请求

- App 发送给设备：

Table 1. 查询设备状态，App 发送数据

数据包结构	内容	说明
Header	Host	mDNS 发现设备，获得设备 IP
	Port	8001
	URL	/dev-state
data	login_id	设备登录名（默认 admin）
	dev_passwd	设备登录密码（默认 12345678）
	user_token	用户 token

- 设备响应：

Table 2. 查询设备状态，设备响应数据

数据包结构	内容	说明
Header	status	200 OK
	Content-Type	application/json
data	isActivated	激活状态(true/false)
	isConnected	云连接状态(true/false)
	version	ROM 版本字符串(如：v0.x.x)

- 实例：
  - App 发送：

```
POST /dev-state HTTP/1.1
Host: 192.168.31.180:8001
Content-Length: 74
Cache-Control: no-cache

{"login_id":"admin","dev_passwd":"12345678","user_token":"11111111"}
```

  - 设备返回：

```
{ "isActivated": true, "isConnected": true, "version": "v0.2.3" }
```

4.2.2 激活请求

- APP 向设备发送：

Table 3. 激活请求，App 发送数据

数据包结构	内容	说明
Header	Host	mDNS 发现设备，获得设备 IP

数据包结构	内容	说明
	Port	8001
	URL	/dev-activate
data	login_id	设备登录名（激活后保存）
	dev_passwd	设置设备登录密码（激活后保存）
	user_token	用户 token

- 设备响应：

Table 4. 激活请求，设备响应数据

数据包格式	内容	说明
Header	status	200 OK
	Content-Type	application/json
data	device_id	设备激活后获得的唯一 ID

- 实例：
  - APP 发送：

```
POST /dev-activate HTTP/1.1
Host: 192.168.31.180:8001
Content-Length: 74
Cache-Control: no-cache
{"login_id":"admin","dev_passwd":"12345678","user_token":"11111111"}
```

  - 设备返回：

```
{ "device_id": "af2b33be/c8934645dd0a" }
```

注意：激活成功后，设备将会保存用户设置的用户名和密码，后续请求会验证该用户名和密码。

4.2.3 用户授权请求

- APP 向设备发送:

Table 5. 用户授权请求，APP 发送数据

数据包结构	内容	说明
Header	Host	mDNS 发现设备，获得设备 IP
	Port	8001
	URL	/dev-authorize
data	login_id	设备登录名
	dev_passwd	设备登录密码
	user_token	用户 token

- 设备响应

Table 6. 用户授权请求，设备响应数据

数据包结构	内容	说明
Header	status	200 OK
	Content-Type	application/json
data	device_id	设备的唯一 ID

- 实例：
  - APP 发送：

```
POST /dev-authorize  HTTP/1.1
Host: 192.168.31.180:8001
Content-Length: 74
Cache-Control: no-cache

{"login_id":"admin","dev_passwd":"12345678","user_token":"22222222"}
```

  - 设备返回：

```
{ "device_id": "af2b33be/c8934645dd0a" }
```

4.2.4 设备注销请求

该方法使得设备从云端注销，下次再使用需要重新激活。

- APP 向设备发送：

Table 7. 设备注销请求，APP 发送数据

数据结构	内容	说明
Header	Host	mDNS 发现设备，获得设备 IP
	Port	8001
	URL	/dev-cloud_reset
data	login_id	设备登录名
	dev_passwd	设备登录密码
	user_token	用户 token

- 设备返回：

Table 8. 设备注销请求，设备返回数据

数据结构	内容	说明
Header	status	200 OK
data	error	{ “error” : <err_code>}

- 实例：
  - APP 发送：

```
POST /dev-cloud_reset  HTTP/1.1
Host: 192.168.31.180:8001
Content-Length: 74
Cache-Control: no-cache

{"login_id":"admin","dev_passwd":"12345678","user_token":"11111111"}
```

- 设备返回：

  成功，无数据实体；

以上操作失败时，返回状态码 500，消息实体返回详细错误码：

{ “error”：<err_code>}

4.2.5 状态码

200: 执行成功

500: 执行失败

详细错误码请参考：MiCOKit 最新 SDK 中，头文件 Common.h 中 MiCO 错误码。

5. 云端消息通信

设备连接上云端后，APP 即可向设备发送消息，详细方法请参考：《MiCO Kit 云数据通信协议》