Espressif IoT Demo 使用手册

版本信息

日期	版本	撰写人	审核人	修改说明
2014.5.13	0.1	巫建刚		草稿

目录

1. 前言

本文主要介绍基于Espressif IoT SDK的嵌入式应用开发，在该IoT Demo中，实现了三类产品：插座、灯、传感器，并且基于外网服务器，实现了对设备的反向控制以及数据的采集。

通过对本文的熟悉，用户可以快速的开发类似应用产品。

2. 总体介绍

2.1. 代码结构

2.1.1. usr 目录

usr 目录下为 IoT Demo 功能实现代码，具体如下：
user_main.c——主入口文件；
user_webserver.c——提供 REST 的轻量 webserver 功能；
user_devicefind.c——提供设备查找功能；
user_esp_platform.c——基于 espressif 服务器的应用功能；
user_json.c——json 包处理功能；
user_plug.c——插座相关功能；
user_light.c——pwm 灯相关功能；
user_humiture.c——温湿度计相关功能；

2.1.2. include 目录

include 目录下问相关头文件，需要注意的是 user_config.h 文件，在该文件中可以对采用平台，以及具体 demo 进行选择，支持如下例子：PLUG_DEVICE，LIGHT_DEVICE，HUMITURE_DEVICE。

2.1.3. driver 目录

driver 当前支持 i2c master，外部按键，pwm，双 uart。

2.2. 工作模式

IoT Demo 中所有应用的 wifi 工作模式均为 softAP->station。

softAP 模式下的 SSID 默认为 ESP_XXXXXX，其中 XXXXXX 为设备 MAC 地址的后面三个字节，默认加密模式为 open。在 softAP 模式下设置需要连接的路由的 ssid 和 password 后，设备即重启进入 station 模式，并自动去连接之前配置的路由。

在 station 模式下，长按按键 5 秒，设备即复位并重启进入 softAP 模式，可重新进行配置。

2.3. 调试工具

IoT Demo 内的 server 采用 REST 架构，PC 端在与 IoT Demo 设备进行通讯时，可采用 curl 命令。

可在如下链接进行指定版本的下载：http://curl.haxx.se/download.html。

3. 局域网功能

softAP 模式默认 ip 为 192.168.4.1，station 模式下 ip 由路由分配。以下 URL 中 ip 信息代表了 softAP 和 station 模式下的 ip。

3.1. 通用功能

3.1.1. 获取版本信息

```
curl -X GET http://ip/client?command=info
```

返回

```
{
    "Version": {
        "major": 0,
        "minor": 6
    },
    "Device": {
        "manufacture", "Espressif Systems"
    }
}
```

3.1.2. 设置连接参数

➢ 设置 station 连接参数

```
curl -X POST -H Content-Type:application/json -d '{"Request":{"Station":{"Connect_Station":{"ssid":"tenda","password":"1234567890","token": "1234567890123456789012345678901234567890"}}}}' http://ip/config?command=wifi
```

设置完成后设备自动重启，进入 station 模式，并自动去连接所设置的路由。

注：

token 字段和 espressif 服务器架构相关，是个随机的长度为 40 的 16 进制数的字符串。

➢ 设置 softAP 参数

```
curl -X POST -H Content-Type:application/json -d '{"Request":{"Softap":{"Connect_Softap":{"authmode":"OPEN","channel":6,"ssid":"ESP_IOT_SOFTAP","password":""}}}}' http://192.168.4.1/config?command=wifi
```

设置 softAP 的参数，设置完后，插座需要重启。

注意：

authmode 支持如下模式：OPEN、WPAPSK、WPA2PSK、WPAPSK/WPA2PSK。

password 长度需不小于 8 个字符。

3.2. 局域网内设备查找

利用在局域网内向端口 1025 发送 UDP 广播包的方法进行设备的查找，发送信息为“Are You Espressif IOT Smart Device?”，设备对在 1025 端口收到的 UDP 广播包进行判断处理，如为该字符串，则做出正确响应，响应内容为：

- 插座

  I’m Plug.xx:xx:xx:xx:xx:xxyyy.yyy.yyy.yyy

- 灯

  I’m Light.xx:xx:xx:xx:xx:xxyyy.yyy.yyy.yyy

- 温湿度

  I’m Humiture.xx:xx:xx:xx:xx:xxyyy.yyy.yyy.yyy

  其中 xx:xx:xx:xx:xx:xx 为开关 MAC 地址，yyy.yyy.yyy.yyy 为开关 ip 地址。

  如不为该字符串，则不做响应。

  可利用网络调试助手来测试该功能。

3.3. 插座

3.3.1. 获取插座状态

```
curl -X GET http://ip/config?command=switch
```

返回

```
{
    "Response": {
        "status": 0
    }
}
```

status 可以为 0 或 1。

3.3.2. 设置插座状态

```
curl -X POST -H Content-Type:application/json -d '{"Response":{"status":1}}'
 http://ip/config?command=switch
```

status 可以为 0 或 1。

3.4. 灯

3.4.1. 获取灯状态

```
curl -X GET http://ip/config?command=light
```

返回

```
{
    "freq": 100,
        "rgb": {
            "red": 100,
            "green": 0,
            "blue": 0
    }
}
```

其中，freq 可以为 1~500，red、green、blue 可以为 0~255。

3.4.2. 设置灯状态

```
curl -X POST -H Content-Type:application/json -d '{"freq":100,"rgb":{"red":2
00,"green":0,"blue":0}}' http://ip/config?command=light
```

其中，freq 可以为 1~500，red、green、blue 可以为 0~255。

3.5. 温湿度

4. 广域网功能

4.1. espressif 服务器平台

关于 espressif 服务器平台的详细使用，会在 espressif 服务器上提供详细的操作及 API 介绍。

4.1.1. 关于设备的 device key

根据 espressif 服务器的架构设计，每台设备均有一个 device key，当前需要预烧到 spi flash 的 0x3f000 位置。

关于设备的 device key，请针对每块开发板，向 espressif 进行申请。后期会在 server 上提供相关用户接口，用户可以自行进行申请。

各个 bin 的烧写顺序： blank.bin->eagle.app.v6.flash.bin->dev_key.bin->eagle.app.v6.irom0text.bin，当不需要重新烧写 blank.bin 和 dev_key.bin 时，可以直接 eagle.app.v6.flash.bin->eagle.app.v6.irom0text.bin。

4.1.2. 激活

- 设备侧

在 softAP 模式下，设置完 ssid、password 及 token 后，设备重启进入 station 模式，在 station 模式下连上路由后，会默认进行一次设备激活。

激活需要往 espressif 的服务器，ip 地址为 114.215.177.97，端口为 8000，发送如下格式的的 tcp 包。

```
{"path": "/v1/device/activate/", "method": "POST", "meta": {"Authorization
": "token HERE_IS_THE_MASTER_DEVICE_KEY"}, "body": {"encrypt_method": "PLAI
N", "bssid": "18:fe:34:70:12:00", "token": "1234567890123456789012345678901
234567890"}}
```

其中的 HERE_IS_THE_MASTER_DEVICE_KEY 为存于 spi flash 的 device key，1234567890123456789012345678901234567890 为在 softAP 模式下写入的 token，需要是个随机值。

响应

```
{"status": 200, "device": {device}, "key": {key}, "token": {token}}
```

- PC 侧

PC 侧在配置完设备的 ssid、password 及 token 后需要连接到一可上外网的路由，进行设备绑定。

```
curl -X POST -H Authorization:token c8922638bb6ec4c18fcf3e44ce9955f19fa3ba12 -d '{"token": "12345678901234567890123456789012345678 90"}' http://114.215.177.97/v1/key/authorize/
```

返回

```
{"status": 200, "key": {"updated": "2014-05-12 21:22:03", "user_id": 1, "product_id": 0, "name": "device activate share token", "created": "2014-05-12 21:22:03", "source_ip": "*", "visibly": 1, "id": 149, "datastream_tmpl_id": 0, "token": "e474bba4b8e11b97b91019e61b7a018cdbaa3246", "access_methods": "*", "is_owner_key": 1, "scope": 3,  "device_id": 29, "activate_status": 1, "datastream_id": 0, "expired_at": "2288-02-22 20:31:47"}}
```

e474bba4b8e11b97b91019e61b7a018cdbaa3246 为获取到的设备 owner key，后面在 PC 侧对设备进行控制时，需要采用这个 key。

4.1.3. 认证

设备激活完成后在连接 espressif 服务器时，需要往 espressif 的服务器，ip 地址为 114.215.177.97，端口为 8000，发送如下格式的的 tcp 包。

```
{"nonce": 12306, "path": "/v1/device/", "method": "GET", "meta": {"Authorization": "token HERE_IS_THE_MASTER_DEVICE_KEY"}}
```

这个 tcp 的包的作用是确认设备自己的身份，每次设备重新连接服务器后的都需要向服务器发送这样一包数据。其中“nonce”是一组随机整数，token 后面是设备的 device key。

服务器收到设备发送的认证包后会向设备回复一个身份确认成功的数据包。

响应

```
{"device": {"status": 2, "updated": "2014-05-10 14:52:59", "activated_at": "2014-05-10 14:52:59", "bssid": "", "last_active": "2014-05-10 14:52:59", "created": "2014-05-08 09:44:57", "description": "description Of device(serial: 47c9a2e8)", "last_pull": "2014-05-10 14:52:59", "visibly": 1, "last_push": "2014-05-10 14:52:59", "name": "name Of device(serial: 47c9a2e8)", "activate_status": 1, "location": "", "is_frozen": 0, "key_id": 134, "serial": "47c9a2e8", "product_id": 1, "id": 29, "is_private": 1, "metadata": "18:fe:34:77:c0:00 temperature"}, "nonce": 12306, "status": 200}
```

认证过程在插座和灯的应用中需要。

4.1.4. PING 服务器

为了保持在长期不进行反向控制设备的情况下，保持 socket 的连接，需要在

每 50s 往 espressif 的服务器，ip 地址为 114.215.177.97，端口为 8000，发送如下格式的的 tcp 包。

```
{"path": "/v1/ping/", "method": "POST","meta": {"Authorization": "token
HERE_IS_THE_MASTER_DEVICE_KEY"}}
```

响应

```
{"status": 200, "message": "ping success", "datetime": "2014-05-12 03:27:26", "nonce
": -1}
```

PING 服务器机制需要在插座及灯这样需要进行反向控制的设备中进行。

4.1.5. 插座

- 设备侧

在进行对设备的反向控制时，存在如下两种情况：

1.设备侧收到服务器发来的get命令时，表示设备需要将自身的状态发送至服务器上，服务器发给设备的get命令格式如下所示：

```
{"body": {}, "nonce": 33377242, "is_query_device": true, "get": {}, "token":
"e474bba4b8e11b97b91019e61b7a018cdbaa3246", "meta": {"Authorization": "token
e474bba4b8e11b97b91019e61b7a018cdbaa3246"}, "path": "/v1/datastreams/plug-
status/datapoint/", "post": {}, "method": "GET"}
```

响应

```
{"status": 200, "datapoint": {"x": 0}, "nonce": 33377242, "is_query_device": true}
```

2.设备侧收到服务器发来的post命令时，表示设备需要改变自身状态，服务器相关的数据包来完成相应的控制动作，如打开开关命令：

```
{"body": {"datapoint": {"x": 1}}, "nonce": 4252886, "is_query_device": true, "get":
{}, "token": "e474bba4b8e11b97b91019e61b7a018cdbaa3246", "meta": {"Authorization":
"token e474bba4b8e11b97b91019e61b7a018cdbaa3246"}, "path": "/v1/datastreams/plug-
status/datapoint/", "post": {}, "method": "POST"}
```

开关完成控制动作后向服务器发送一个更新状态成功的响应，格式如下，响应回复的 nonce 后面的值必须后服务器先前发送的控制命令中的 nonce 值一致，以表示每次控制和回应相互对应。

响应

```
{"status": 200, "datapoint": {"x": 1}, "nonce": 4252886, "is_query_device": true}
```

- PC 侧

获取插座状态

```
curl -x GET -H Content-Type:application/json Authorization: token
e474bba4b8e11b97b91019e61b7a018cdbaa3246 http://114.215.177.97/v1/datastreams/plug-status
/datapoint
```

响应

```
{"status": 200, "nonce": 11432809, "datapoint": {"x": 1}, "is_query_device": true}
```

设置插座状态

```
curl -x POST -H Content-Type:application/json Authorization: token
e474bba4b8e11b97b91019e61b7a018cdbaa3246 –d '{"datapoint":{"x":1}}, "meta":
{"Authorization": "token e474bba4b8e11b97b91019e61b7a018cdbaa3246"}'
http://114.215.177.97/v1/datastreams/plug-status/datapoint
```

响应

```
{"status": 200, "nonce": 11432809, "datapoint": {"x": 1}, "is_query_device": true}
```

4.1.6. 灯

✧ 设备侧

在进行对设备的反向控制时，存在如下两种情况：

1.设备侧收到服务器发来的get命令时，表示设备需要将自身的状态发送至服务器上，服务器发给设备的get命令格式如下所示：

```
{"body": {}, "nonce": 8968711, "is_query_device": true, "get": {}, "token":
"e474bba4b8e11b97b91019e61b7a018cdbaa3246", "meta": {"Authorization": "token
e474bba4b8e11b97b91019e61b7a018cdbaa3246"}, "path":
"/v1/datastreams/light/datapoint/", "post": {}, "method": "GET"}
```

响应

```
{"nonce": 5619936, "datapoint": {"x": 100,"y": 200,"z": 0,"k": 0,"l": 50},"is_query_
device": true}
```

2.设备侧收到服务器发来的post命令时，表示设备需要改变自身状态，服务器相关的数据包来完成相应的控制动作，如打开开关命令：

```
{"body": {"datapoint": {"y": 200, "x": 100, "k": 0, "z": 0, "l": 50}}, "nonce":
5619936, "is_query_device": true, "get": {}, "token":
"e474bba4b8e11b97b91019e61b7a018cdbaa3246 ", "meta": {"Authorization": "token
e474bba4b8e11b97b91019e61b7a018cdbaa3246"}, "path":
"/v1/datastreams/light/datapoint/", "post": {}, "method": "POST"
```

响应

```
{"nonce": 5619936, "datapoint": {"x": 100,"y": 200,"z": 0,"k": 0,"l": 50},"is_query_
device": true}
```

其中 X 表示 freq 可以为 1~500，Y，Z，K 表示灯的不同颜色可以为 0~255 具体 Y 表示 red、Z 表示 green、K 表 blue。L 参数目前保留。

✧ PC 侧

获取灯状态

```
curl -x GET -H Content-Type:application/json Authorization: token
e474bba4b8e11b97b91019e61b7a018cdbaa3246 http://114.215.177.97/v1/datastreams/light/datapoint
```

响应

```
{"nonce": 5619936, "datapoint": {"x": 100,"y": 200,"z": 0,"k": 0,"l": 50},"is_query_
device": true}
```

设置灯状态

```
curl -x POST -H Content-Type:application/json Authorization: token
e474bba4b8e11b97b91019e61b7a018cdbaa3246 –d '{"datapoint":{"x": 100, "y": 200, "z":
0, "k": 0,"l": 50}} ' http://114.215.177.97/v1/datastreams/light/datapoint
```

返回

```
{"nonce": 5619936, "datapoint": {"x": 100,"y": 200,"z": 0,"k": 0,"l": 50},"is_query_
device": true}
```

其中 X 表示 freq 可以为 1~500，Y，Z，K 表示灯的不同颜色可以为 0~255 具体 Y 表示 red、Z 表示 green、K 表 blue。L 参数目前保留。

4.1.7. 温湿度

✧ 设备侧

数据包上传，格式：

```
{"nonce": 1, "path": "/v1/datastreams/tem_hum/datapoint/", "method": "POST",
"body": {"datapoint": {"x": 35,"y": 32}}, "meta": {"Authorization": "token
HERE_IS_THE_MASTER_DEVICE_KEY"}}
```

温度值使用 X 来表示，湿度值使用 y 来表示，token 后面携带设备的 device key，数据上传成功后服务器会给设备上传成功的响应。

响应

```
{"status": 200, "datapoint": {"updated": "2014-05-14 18:42:54", "created": "2014-05-
14 18:42:54", "visibly": 1, "datastream_id": 16, "at": "2014-05-14 18:42:54", "y": 3
2, "x": 35, "id": 882644}}
```

响应信息中携带数据更新的最后时间。

✧ PC 侧

PC 侧通过如下两类 API 来获得传感器的数据，红色字体是用户的 ownkey，

1。1.获得最新的温湿度信息：

```
curl -x GET -H Content-Type:application/json Authorization: token
e474bba4b8e11b97b91019e61b7a018cdbaa3246 http://114.215.177.97/v1/datastreams/
tem_hum /datapoint
```

2.获得传感器采集的历史数据，

```
curl -x GET -H Content-Type:application/json Authorization: token
e474bba4b8e11b97b91019e61b7a018cdbaa3246 http://114.215.177.97/v1/datastreams/
tem_hum /datapoints
```