

一起学习 CC3200 系列教程之 Yeelink 传数据

阿汤哥

序:

能力有限，难免有错，有问题请联系我，

QQ1519256298 hytga@163.com

Pdf 下载 <http://pan.baidu.com/s/1hqiWB56>

为了提高博客的活跃，，如果你需要源代码，请留下邮箱，，，谢谢大家啊，，不然才两个评论，让我情何以堪啊

Yeeklink: 作为一个开放的公共物联网接入平台，目的是为服务所有所有的爱好者和开发者，使传感器数据的接入、存储和展现变得轻松简单。

TCP:一般性的建立 TCP 连接需要两个参数：IP 和端口。这里我们使用 dns 获取其 ip，


HTTP: web 浏览器使用的就是 HTTP 协议，这个是建立在 tcp 的基础上，本文只是简单地模仿了 HTTP 的格式，

Socket: 怎么建立 TCP 连接，就需要用到 socket，这个很像 linux 的 socket。

Json: json 是一种数据的传输格式，譬如 `{"name":阿汤哥, "age":100 }` 这个的数据的含义是：名字叫阿汤哥，年龄是 100 岁，，简单明了的协议。

STA：当 CC3200 需要连上路由器的 wifi 的时候就叫 STA 模式，，当用电脑连上 CC3200 的 wifi 就叫 AP 模式。

yeelink 上的各种截图及结果截图

**管理API Key**
管理首页 > API Key

ⓘ 本操作不可恢复，可能会造成您的设备运行不正常，请联系客服或确认。

API Key

🔍 5d4c9fb6b2ba9386d497e6cf6a2a9f75



暂无图片

TEST PUB
专用于测试

设备ID：20555

设备地址：<http://www.yeelink.net/devices/20555>

API 地址：<http://api.yeelink.net/devices/20555>

部署

编辑

添加传感器

输入传感器名搜索..... 🔍

**TestNum** 数据类型 传感器id <http://blog.csdn.net/hytgab> 打开图表

传感器ID：38264

地址：http://www.yeelink.net/devices/20555/#sensor_38264

API 地址：<http://api.yeelink.net/v1.1/device/20555/sensor/38264/datapoints>

编辑 删除 ▶ 触发动作(0)

110

摄氏度 / °C

**TestSwitch** 开关型

传感器ID：38265

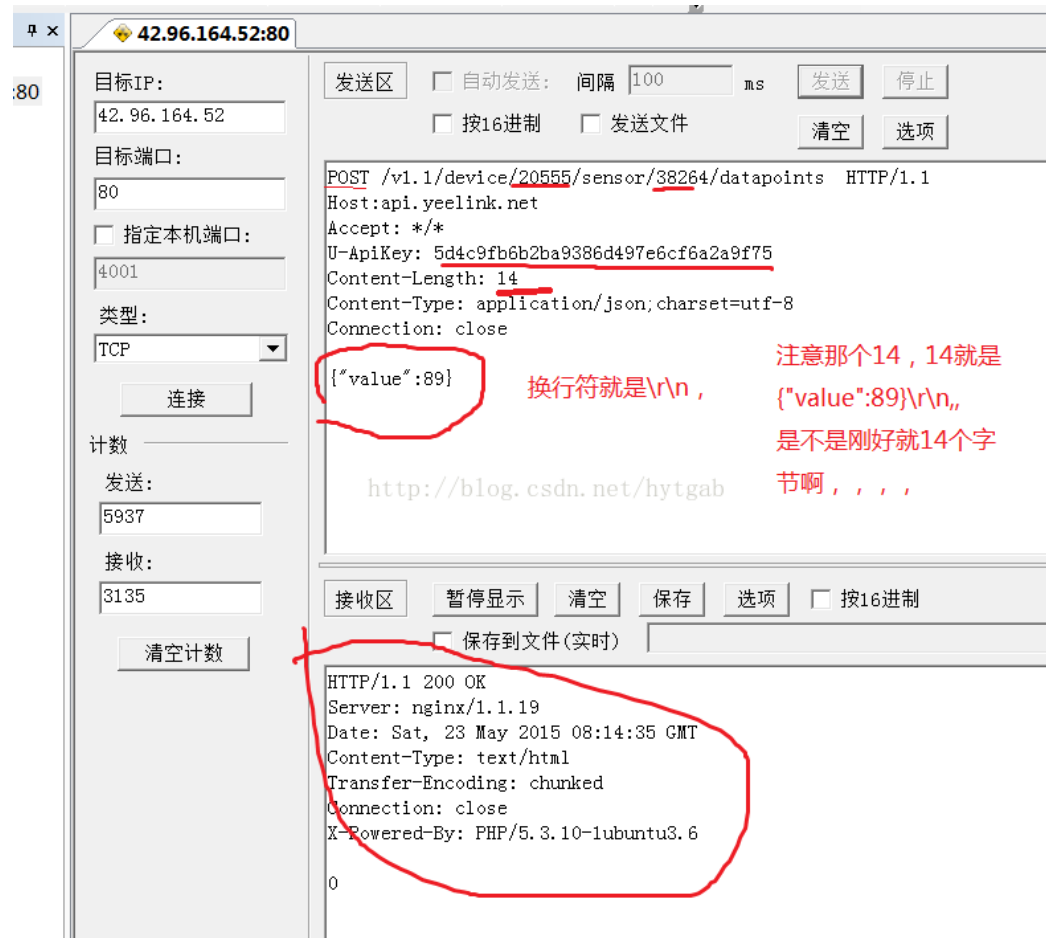
地址：http://www.yeelink.net/devices/20555/#sensor_38265

API 地址：<http://api.yeelink.net/v1.1/device/20555/sensor/38265/datapoints>

编辑 删除



直接使用 tcp 测试工具把数据上传，
post 上传数据



get, 获取数据

42.96.164.52:80

目标IP: 42.96.164.52
目标端口: 80
☐ 指定本机端口: 4001
类型: TCP
连接

发送区 ☐ 自动发送: 间隔 100 ms 发送 停止
☐ 按16进制 ☐ 发送文件 清空 选项

设备号地址信息传感器地址信息,
GET /v1.1/device/20555/sensor/38265/datapoints HTTP/1.1
Host: api.yeelink.net
Accept: */*
U-ApiKey: 5d4c9fb6b2ba9386d497e6cf6a2a9f75 key号
Content-Length: 0
Connection: close

printf get done 其实这句话是没有的,但是没有这句话却会获取失败
什么原因: 可能是http协议的关系,需要在Connection: close后面加两个换行符

http://blog.csdn.net/hytagab

接收区 暂停显示 清空 保存 选项 ☐ 按16进制
☐ 保存到文件(实时)

HTTP/1.1 200 OK
Server: nginx/1.1.19
Date: Sat, 23 May 2015 08:08:24 GMT
Content-Type: text/html
Transfer-Encoding: chunked
Connection: close
X-Powered-By: PHP/5.3.10-1ubuntu3.6

55 接收的有效数据, JSON格式
{ "value": 1, "timestamp": "2015-05-23T14:28:56", "sensor_id": "38265", "device_id": "20555" }

结果:



软件流程:

首先, CC3200 作为了一个 STA, 通过路由器的网络连上因特网, 使用 DNS 获取 Yeelink 的 ip, Yeelink 的端口号是 80 端口, 一般 80 端口是用作 HTTP 的, 我们模仿 HTTP 的协议把数据传到 Yeelink 上, 数据的传输格式是 JSON 格式。

首先, 需要在 common 找到如下的定义, 并根据你的网络情况进行更改

```
#define SSID_NAME "需要更改" /* AP SSID */
#define SECURITY_TYPE SL_SEC_TYPE_WPA_WPA2/* 加密类型, 一般的都是这个Security type (OPEN or WEP or WPA*/
#define SECURITY_KEY "需要更改" /* Password of the secured AP */
```

```
#define SSID_LEN_MAX      32
#define BSSID_LEN_MAX     6
```

你的 yeelink 的设备号等相关信息：

```
//yeelink 的设备地址，需要根据你的情况进行更改
#define mainYEELINK_DEVICE_ID      20555
//yeelink 的传感器地址，需要根据你的情况进行更改
#define mainYEELINK_TestNum_ID     38264
//yeelink 的 key 号，需要根据你的情况进行更改
#define mainYEELINK_API_KEY
    "5d4c9fb6b2ba9386d497e6cf6a2a9f75"

//设备地址，需要根据你的情况进行更改
#define mainYEELINK_TestSwitch_ID  38265
```

HTTP 的数据

```
//http 的 post 数据格式，不需要更改
char g_cPostData[] = "POST /v1.1/device/%d/sensor/%d/datapoints
HTTP\1.1\r\n"
    "Host:api.yeelink.net\r\n"
    "Accept: */*\r\n"
    "U-ApiKey: %s \r\n"
    "Content-Length: %d \r\n"
    "Content-Type: application/json;charset=utf-8\r\n"
    "Connection: close\r\n"
    "\r\n"
    "%s";

//http 的 get 数据格式
char g_cGetData[] = "GET /v1.1/device/%d/sensor/%d/datapoints
HTTP\1.1\r\n"
    "Host:api.yeelink.net\r\n"
    "Accept: */*\r\n"
    "U-ApiKey: %s \r\n"
    "Content-Length: 0 \r\n"
    "Connection: close\r\n"
    "\r\n\r\n";
```

我是直接使用官方的 `httpserver` 的例程进行更改的，设置是：使用 `freertos` 系统
创建任务，在 `main` 函数中更改

```
//  
// Create HTTP Server Task  
//  
lRetVal = osi_TaskCreate(vYeelinkTask, (signed  
char*)"vYeelinkTask",  
OSI_STACK_SIZE, NULL, OOB_TASK_PRIORITY,  
NULL );
```

任务：连接上网络，获取 yeelink 的 ip，发送 post 数据和 get 数据

post 数据：就是把你的数据传输到 yeelink 上，可以实现的功能：上传传感器的值

get 数据：就是把 yeelink 的数据传到 CC3200，可以实现的功能：远程遥控家里的设备

```
static void vYeelinkTask(void *pvParameters) {  
    unsigned long ip,temp = 0;  
    char buffer[20];  
    UART_PRINT("success:%s %d\r\n",__FUNCTION__,__LINE__);  
    scNetStaInit();  
    delay();  
    while(1) {  
        ip = lGetYeelinkIp();  
        if (ip != 0) {  
            //这个主要就是实现 JSON 的数据  
            //{"value":12}\r\n  
  
            snprintf(buffer, 20, "{ \"value\":%d} \r\n", temp++);  
            //这个主要就是合成 HTTP 协议  
            //这个具体可以看 g_cPostData  
  
            snprintf(g_buffer, 300, g_cPostData, mainYEELINK_DEVICE_ID, mainY  
EELINK_TestNum_ID, mainYEELINK_API_KEY, strlen(buffer), buffer);  
            UART_PRINT("\r\n Send:\r\n%s\r\n", g_buffer);  
            //发送数据， 参数： 数据， ， 数据长度， ip， 端口  
  
            vSendOneDataWithIpPort(g_buffer, strlen(g_buffer), ip, mainYEELI  
NK_PORT);  
        }
```

```

        delay();
        delay();

        snprintf(g_buffer, 300, g_cGetData, mainYEELINK_DEVICE_ID, mainYEELINK_TestSwitch_ID, mainYEELINK_API_KEY);
        UART_PRINT("\r\n Send:\r\n%s\r\n", g_buffer);

        vSendOneDataWithIpPort(g_buffer, strlen(g_buffer), ip, mainYEELINK_PORT);

        delay();
        delay();
    }
    delay();
}
}

```

连接网络

```

//设置 sta 模式并连接到网络
//返回 0 ， 连接成功
//返回-1， 连接失败
static signed char scNetStaInit(void) {
    SlSecParams_t secParams = {0};
    unsigned short len, config_opt;
    long lRetVal = -1;
    //主要是设置成默认的状态, 这个在 httpserver 就有了
    lRetVal = ConfigureSimpleLinkToDefaultState();
    if(lRetVal < 0)
    {
        if (DEVICE_NOT_IN_STATION_MODE == lRetVal)
            UART_PRINT("Failed to configure the device in its default state\n\r");

        LOOP_FOREVER();
    }
    //需要先调用这个
}

```

```

lRetVal = sl_Start(0, 0, 0);
if (lRetVal < 0)
{
    UART_PRINT("Failed to start the device \n\r");
    LOOP_FOREVER();
}
// staring simplelink
g_uiSimplelinkRole = sl_Start(NULL, NULL, NULL);

// 设置成 sta 模式
if(g_uiSimplelinkRole != ROLE_STA )
{
    //Switch to STA Mode
    lRetVal = sl_WlanSetMode(ROLE_STA);
    ASSERT_ON_ERROR(lRetVal);

    lRetVal = sl_Stop(SL_STOP_TIMEOUT);

    g_usMCNetworkUstate = 0;
    g_uiSimplelinkRole = sl_Start(NULL, NULL, NULL);
}
//密码
secParams.Key = (signed char*)SECURITY_KEY;
//密码长度
secParams.KeyLen = strlen(SECURITY_KEY);
//加密类型
secParams.Type = SECURITY_TYPE;
//进行连接，参数有 SSID ， 密码
lRetVal = sl_WlanConnect((signed char*)SSID_NAME,
strlen(SSID_NAME), 0, &secParams, 0);
ASSERT_ON_ERROR(lRetVal);
//判断有没有连接成功。其实就是判断 g_ulStatus 响应的位有没有设置成功
功
//g_ulStatus 的值在 SimpleLinkWlanEventHandler 这个函数内被更改
//SimpleLinkWlanEventHandler 是一个回调函数，主要的就是作用就是
//当 CC3200 连上网络或者断开网络的时候会被调用
while((!IS_CONNECTED(g_ulStatus)) ||
(!IS_IP_ACQUIRED(g_ulStatus)))

```



```

    {
        //进行循环，等待成功

    }
    return 0;
}

```

获取 yeelink 网址对应的 ip 地址，使用的是 dns 服务

```

//获取 yeelink 网址对应的 ip 地址
unsigned long lGetYeelinkIp(void) {
    unsigned long ip;
    signed int retval;
    retval = Network_IF_GetHostIP(mainYEELINK_SITE, &ip);
    if (retval < 0) {
        return 0;
    } else {
        return ip;
    }
}

//这个函数是我从 network_if 文件拷贝出来的. 不能直接包含那个文件
//因为会引起重复定义的错误. 所以你直接
long Network_IF_GetHostIP( char* pcHostName, unsigned long * pDestinationIP )
{
    long lStatus = 0;
    //利用这个函数就可以把 www. yeelink. net 转成 IP, ,
    //当然你也可以不用这个函数, , 直接把 IP 规定住,
    //怎么手动查 www. yeelink. net:    win 系统下, ping 一下这个 www. yeelink. net 网址就会出现 ip 信息
    lStatus = sl_NetAppDnsGetHostByName((signed char *) pcHostName,
                                          strlen(pcHostName),
                                          pDestinationIP,
SL_AF_INET);
    ASSERT_ON_ERROR(lStatus);

    UART_PRINT("Get Host IP succeeded. \n\rHost: %s IP: %d.%d.%d.%d \n\r\n\r",
              pcHostName, SL_IPV4_BYTE(*pDestinationIP, 3),
              SL_IPV4_BYTE(*pDestinationIP, 2),

```

```

        SL_IPV4_BYTE(*pDestinationIP, 1),
        SL_IPV4_BYTE(*pDestinationIP, 0));

    return lStatus;

}

```

发送数据

```

char g_buffer[300];
//发送数据给指定的 ip 和端口号，只有一次
//执行过程
//1、新建连接
//2、发送数据、
//3、断开连接
void vSendOneDataWithIpPort(unsigned char *data, unsigned long
len, unsigned long ip, unsigned long port) {

    SockAddrIn_t  sAddr;
    int            iAddrSize;
    int            iSockID;
    int            iStatus;
    int count = 0, size;
    //filling the TCP server socket address
    //使用的是 socket
    //这个我忘了，基本是固定不变
    sAddr.sin_family = SL_AF_INET;
    //端口信息，
    sAddr.sin_port = sl_Htons((unsigned short)port);
    //ip 信息
    sAddr.sin_addr.s_addr = sl_Htonl((unsigned int)ip);

    iAddrSize = sizeof(SockAddrIn_t);

    // creating a TCP socket
    //创建一个 tcp 连接
    iSockID = sl_Socket(SL_AF_INET, SL_SOCKET_STREAM, 0);
    if( iSockID < 0 )
    {

```

```

        UART_PRINT("Error:%s %d\r\n", __FUNCTION__, __LINE__);
    }

    // connecting to TCP server
    //连接到 tcp 的服务器，参数：socket 的句柄，目的地址信息
    iStatus = sl_Connect(iSockID, ( SlSockAddr_t *)&sAddr,
iAddrSize);
    if( iStatus < 0 )
    {
        // error
        iStatus = sl_Close(iSockID);
        if (iStatus < 0)
            UART_PRINT("Error:%s %d\r\n", __FUNCTION__, __LINE__);
    }

    //连接完成最好不要直接就发送数据，经我的测试是，马上发送数据可能会成功，但是数据却没有传过去
    delay();

    //发送数据，参数：socket 的句柄，数据指针，数据长度，最后一个参数没有作用
    iStatus = sl_Send(iSockID, data, len, 0 );
    if( iStatus <= 0 )
    {
        // error
        iStatus = sl_Close(iSockID);
        if (iStatus < 0)
            UART_PRINT("Error:%s %d\r\n", __FUNCTION__, __LINE__);
    }

    //接收数据。，参数：socket 的句柄，缓冲区，缓冲器大小，返回的是接收的数据大小
    //请注意这个函数可能是阻塞的
    //阻塞的意思就是，会一直等到有数据才返回，或者有问题才返回
    size = sl_Recv(iSockID, g_buffer, 300, 0);
    transHeader = (SlTransceiverRxOverHead_t *)g_buffer;
    g_buffer[size] = 0;
    UART_PRINT("\r\nReceive:\r\n%s\r\n", g_buffer);
    //关闭连接
    iStatus = sl_Close(iSockID);
    if (iStatus < 0)

```

```
        UART_PRINT("Error:%s %d\r\n", __FUNCTION__, __LINE__);  
    }
```

到这里基本上所有的代码都已经上传完毕,,