

SV6X66 系列

快速启动连接方案

关于本手册

本手册介绍了 SV6X66 系列的快速启动连接方案

发布说明：

版本	日期	编辑	描述
V1.0	2020/02/17	Andy	初始版本

文档变更通知&证书下载：

本文档更新不会逐一通知，用户需要使用时请自行去南方硅谷官网上下载最新版资料；需要相关证书的用户请联系南方硅谷客服 请知悉！

目录

1. 前言	4
2. 优化步骤	4
3. 测试方法	4
4. 流程说明	5
5. 测试数据	5
5.1. 开放环境测试	5
5.2. 屏蔽房测试	5
6. 代码分析	6

1. 前言

IOT 产品在一些具体的应用中，需要有快速启动连接路由器的功能。例如体脂称方案需要在 10 秒内开机上报数据，照明灯开机等。SV6x66 为快速启动连接提供完整的方案，其中冷启动与热启动的开机时间一样，150ms 以内可完成重连功能。

2. 优化步骤

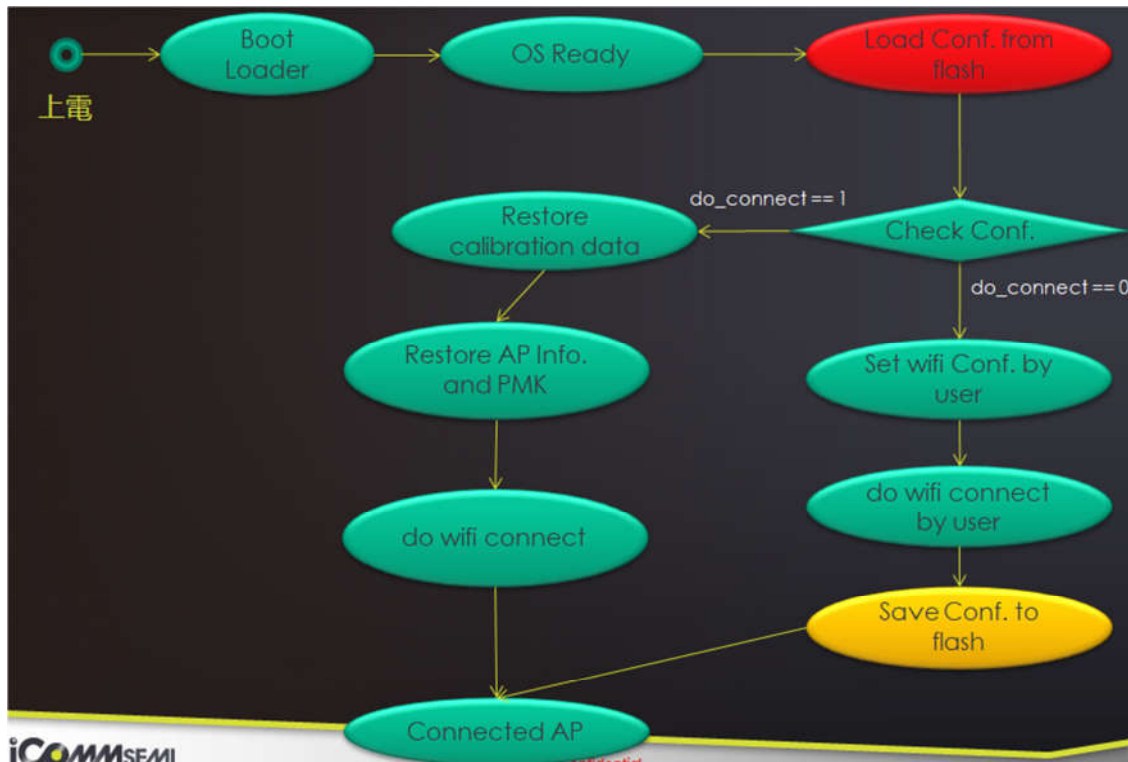
- PMK 保存与恢复
 - 每个加密的路由器，连接成功后都会成一个 PMK 值。当路由器的 SSID 与 PSK 没有改变时，PMK 值不会变化。所以需要把 PMK 值保存到 Flash 中，下次开机时直接使用 PMK 值，减少 PMK 的计算时间
- 校准值的保存与恢复
 - 与 PMK 的处理一致，当重连时读取 Flash 的校准值。
- LOG 信息打印
 - printf 打印比较耗时，开机过程需要关闭其 LOG
- 关闭 Bootloader Uart 升级功能
- 取消 DHCP ARP 检查
 - ARP 指检查局域网内时间有同样的 IP 地址存在，设备都是 DHCP 的情况，不会产生 IP 地址冲突的情况，所以此检查可以关闭

3. 测试方法

- 编译/烧录 fast_connect 工程（USB 或者 SPI 烧录才会更新 bootloader，才能关闭 Bootloader Uart 升级功能）

- 连接指定的 AP, 例如 AT+FASTCONNECT=wifi-46089c-1,90091000
 - 连接成功后, 重新上电或 reset, 观察启动/连接时间
- 1、当信号不好时, Wi-Fi 数据包丢失并多次重传, 导致连接时间过长
- 2、AT+SET_FASTCONNECT =enable/disable 可以开启/关闭重连

4. 流程说明



5. 测试数据

5.1. 开放环境测试

Stage	Test 1	Test 2	Test 3	Test 4	Test 5
platform ready	53 ms	53 ms	53 ms	53 ms	53 ms
Wifi connect	34 ms	40 ms	41 ms	28 ms	40 ms
Dhcp stage	50 ms	57 ms	51 ms	48 ms	41 ms
Total	137 ms	150 ms	145 ms	129 ms	134 ms

5.2. 屏蔽房测试

Stage	Test 1	Test 2	Test 3	Test 4	Test 5
platform ready	53 ms	53 ms	53 ms	53 ms	53 ms
Wifi connect	24 ms	22 ms	27 ms	22 ms	22 ms
Dhcp stage	38 ms	39 ms	38 ms	39 ms	38 ms
Total	115 ms	114 ms	118 ms	114 ms	113 ms

6. 代码分析

文件: projects/fast_connect/src/app/main.c

函数名: APP_Init

从 Flash 中读取保存的 ap_record 结构体, 如果信息有效, 则读取 cal_result 校准恢复

```

254 rf_table_init();
255
256 cfg_user_read_cfg((uint8_t*)&ap_record, sizeof(ap_record));
257 if(ap_record.magic_num == 0x20767373 && ap_record.do_connect == 1)
258 {
259     memcpy(&cal_result, &ap_record.backup_cal_result, sizeof(cal_result));
260 }
261
262 WIFI_INIT();
263 #ifdef TCPIPSTACK_EN
264 net_stack_init(NULL);

```

把保存的信息, 恢复至 ap_list 结构体中, 然后创建 wifi_connect_task 任务进行连接

```

254
255 if(ap_record.magic_num == 0x20767373)
256 {
257     if(ap_record.do_connect == 1)
258     {
259         rf_set_ch(ap_record.channel, 0, NL80211_CHAN_HT20);
260
261         memset(&ap_list[getAvailableIndex()], 0, sizeof(TAG_AP_INFO));
262         ap_list[getAvailableIndex()].channel = ap_record.channel;
263         memcpy(ap_list[getAvailableIndex()].mac, ap_record.mac_addr, 6);
264         memcpy(ap_list[getAvailableIndex()].name, ap_record.ssid, ap_record.ssid_len);
265         ap_list[getAvailableIndex()].name_len = ap_record.ssid_len;
266         gwifistatus.available_index++;
267         set_wifi_config((u8*)ap_record.ssid, ap_record.ssid_len, (u8*)ap_record.key, ap_record.key_len, NULL, 0);
268
269         OS_TaskCreate(wifi_connect_task, "wifi_connect", 512, NULL, OS_TASK_LOW_Prio, NULL);
270     }
271 }

```

函数名: fastwificallback

连接成功后, 保存信息至 Flash 中

```

88     struct wifi_record new_record;
89     memset(&new_record, 0, sizeof(new_record));
90
91     new_record.magic_num = 0x20767373;
92     new_record.do_connect = 1;
93     new_record.channel = gwifistatus.connAP[idx].channel;
94     new_record.ssid_len = gwifistatus.connAP[idx].ssid_len;
95
96     new_record.security_type = gwifistatus.connAP[idx].security_type;
97     new_record.security_subType = gwifistatus.connAP[idx].security_subType;
98     new_record.key_len = gwifistatus.connAP[idx].key_len;
99
100    memset(&new_record.ssid, 0, 32);
101    memcpy(&new_record.ssid, gwifistatus.connAP[idx].ssid, new_record.ssid_len);
102    memset(&new_record.key, 0, 64);
103
104    if( new_record.security_type != 0 && new_record.security_type != 1) //WPA2
105    {
106        for(i=0; i<32; i++)
107        {
108            new_record.key[i*2] = hex2char( (gwifistatus.connAP[0].pmk[i]&0xf0) >> 4);
109            new_record.key[i*2+1] = hex2char(gwifistatus.connAP[0].pmk[i]&0x0f);
110        }
111        new_record.key_len = 64;
112    }
113    else //Open or WEP
114    {
115        memcpy(&new_record.key, gwifistatus.connAP[idx].key, gwifistatus.connAP[idx].key_len);
116    }
117

```

注意, WPA2 与 Open or WEP 处理的方法有区别