XR806 WLAN 冷启动快连方案

开发指南

版本号：1.0

发布时间：2021-02-23

深圳市索智科技有限公司SuoZhi

版本历史

版本	日期	责任人	版本描述
1.0	2021-02-23	AWA 1029	创建文档。

深圳市索智科技有限公司SuoZhi

版权所有©广州芯之联科技有限公司。保留一切权利

目录

版本历史 i
目录 ii
表格目录 iv
1 前言 1
1.1 文档简介 1
1.2 目标读者 1
1.3 适用范围 1
1.4 文档约定 1
1.4.1 标志说明 1
1.4.2 地址与数据描述方法约定 1
1.4.3 数值单位约定 2
2 概述 3
2.1 背景说明 3
2.2 规格特性 3
2.3 文件位置 3
3 技术说明 4
3.1 功能实现 4
3.1.1 实现流程 4
3.1.2 保存信息 4
3.2 启动时间 5
3.2.1 启动时间记录 5
3.2.2 影响因素 5
3.2.2.1 串口打印 5
3.2.2.2 CPU 频率 5
3.2.2.3 Flash ctrl 时钟 5
4 应用说明 6
4.1 应用简述 6
4.2 配置说明 6
4.3 接口说明 6
4.3.1 WLAN 连接操作接口 7
4.3.1.1 wlan_sta_set 7

版权所有©广州芯之联科技有限公司。保留一切权利

4.3.1.2 wlan_sta_enable 7
4.3.1.3 wlan_sta_gen_psk 8
4.3.2 sysinfo 操作接口 8
4.3.2.1 sysinfo_get 8
4.3.2.2 sysinfo_save 9
4.3.3 WLAN BSS 信息操作接口 10
4.3.3.1 wlan_sta_get_bss_size 10
4.3.3.2 wlan_sta_get_bss 10
4.3.3.3 wlan_sta_set_bss 11
4.4 指令说明 11
4.4.1 连接配置指令 11
4.4.2 连接使能指令 11
4.4.3 清除保存的 BSS 信息指令 12
5 示例说明 13
5.1 示例简介 13
5.1.1 获取方式 13
5.1.2 准备工作 13
5.1.3 操作步骤 13
5.2 关键实现 14
5.3 效果展示 18

深圳市索智科技有限公司SuoZhi

版权所有©广州芯之联科技有限公司。保留一切权利

表格目录

表 2-1 冷启动快连方案规格特性表.....3
表 2-2 WLAN 冷启动快连的文件位置.....3
表 3-1 WLAN 冷启动快连需要保存的信息.....4
表 4-1 XR806 WLAN 模块工程配置说明.....6
表 4-2 WLAN 冷启动快连接口简介.....6
表 4-3 wlan_sta_set 接口函数说明.....7
表 4-4 wlan_sta_enable 接口函数说明.....7
表 4-5 wlan_sta_gen_psk 接口函数说明.....8
表 4-6 wlan_gen_psk_param_t 结构体的成员说明.....8
表 4-7 sysinfo_get 接口函数说明.....8
表 4-8 sysinfo 结构体的成员说明.....9
表 4-9 sysinfo_netif_param 结构体的成员说明.....9
表 4-10 sysinfo_save 接口函数说明.....9
表 4-11 wlan_sta_get_bss_size 接口函数说明.....10
表 4-12 wlan_sta_get_bss 接口函数说明.....10
表 4-13 wlan_sta_bss_info_t 结构体的成员说明.....10
表 4-14 wlan_sta_set_bss 接口函数说明.....11

深圳市索智科技有限公司SuoZhi

版权所有©广州芯之联科技有限公司。保留一切权利

1 前言

1.1 文档简介

本文档主要介绍 WLAN 冷启动快连方案的基本原理和使用方法，并说明使用该模块的注意事项，以方便开发者正确使用该模块进行开发。

1.2 目标读者

XR806 SDK 用户、WLAN 开发使用人员。

1.3 适用范围

此文档适用于 XR806 SDK，支持 XR806 系列芯片产品。

1.4 文档约定

1.4.1 标志说明

本文档采用各种醒目的标志来表示在操作过程中应该特别注意的地方，这些标志的含义如下：

标识	说明
警告	该标志后的说明应给予格外关注，如果不遵守，可能会导致人员受伤或死亡。
注意	提醒操作中应注意的事项。不当的操作可能会损坏器件，影响可靠性、降低性能等。
说明	为准确理解文中指令、正确实施操作而提供的补充或强调信息。
窍门	一些容易忽视的小功能、技巧。了解这些功能或技巧能帮助解决特定问题或者节省操作时间。

1.4.2 地址与数据描述方法约定

本文档在描述地址、数据时遵循如下约定：

符号	例子	说明
0x	0x0200，0x79	地址或数据以 16 进制表示。
0b	0b010，0b00 000 111	数据采用二进制表示(寄存器描述除外)。
X	00X，XX1	数据描述中，X 代表 0 或 1。 例如，00X 代表 000 或 001；XX1 代表 001，011，101 或 111。

版权所有©广州芯之联科技有限公司。保留一切权利

1.4.3 数值单位约定

本文档在描述数据容量（如 NAND 容量）时，单位词头代表的是 1024 的倍数；描述频率、数据速率等时则代表的是 1000 的倍数。具体如下：

类型	符号	对应数值
数据容量（如 NAND 容量）	1 K	1024
	1 M	1 048 576
	1 G	1 073 741 824
频率，数据速率等	1 k	1000
	1 M	1 000 000
	1 G	1 000 000 000

版权所有©广州芯之联科技有限公司。保留一切权利

2 概述

2.1 背景说明

冷启动快连（Cold Start Fast Connection）是 WLAN 模块用于适配一些特定的应用场景所使用的功能，其主要目的是缩短设备从冷启动到连接上 AP 过程的时间，使设备在极低功耗的情况下可以迅速连接上 AP 进行通信。

2.2 规格特性

冷启动快连方案提供了以下规格特性。

表 2-1 冷启动快连方案规格特性表

规格特性	规格描述	备注
支持保存 AP 信息	支持保存 AP 的 SSID 和计算后的十六进制 PSK 值，以及 AP 的信道、加密方式等信息	该信息保存在 Flash 的指定位置
支持保存 IP 地址	支持保存首次连接后获取的 IP 地址，后续均使用静态 IP 进行连接	该信息保存在 sysinfo 中
支持修改 AP 信息	支持完成快连流程后，将 AP 信息清除，然后重新进行连接，并保存新的 AP 信息	AP 信息清除后，需要重新进行一次普通连接
支持启动时间统计	支持冷启动的各个时间段的耗时统计	

2.3 文件位置

以 SDK 包为根目录，WLAN 冷启动快连涉及到的主要文件位置如下。

表 2-2 WLAN 冷启动快连的文件位置

组件名	文件分类	文件位置
WLAN 冷启动快连	库文件	./lib/xradio_v3/libwlan.a
	头文件	./include/net/wlan/wlan.h ./include/net/wlan/wlan_defs.h
	示例工程	./project/example/wlan_csfc

说明

XR806 SDK 可在以下 GitHub 仓库获取：https://github.com/XradioTech/xr806_sdk.git

版权所有©广州芯之联科技有限公司。保留一切权利

3 技术说明

3.1 功能实现

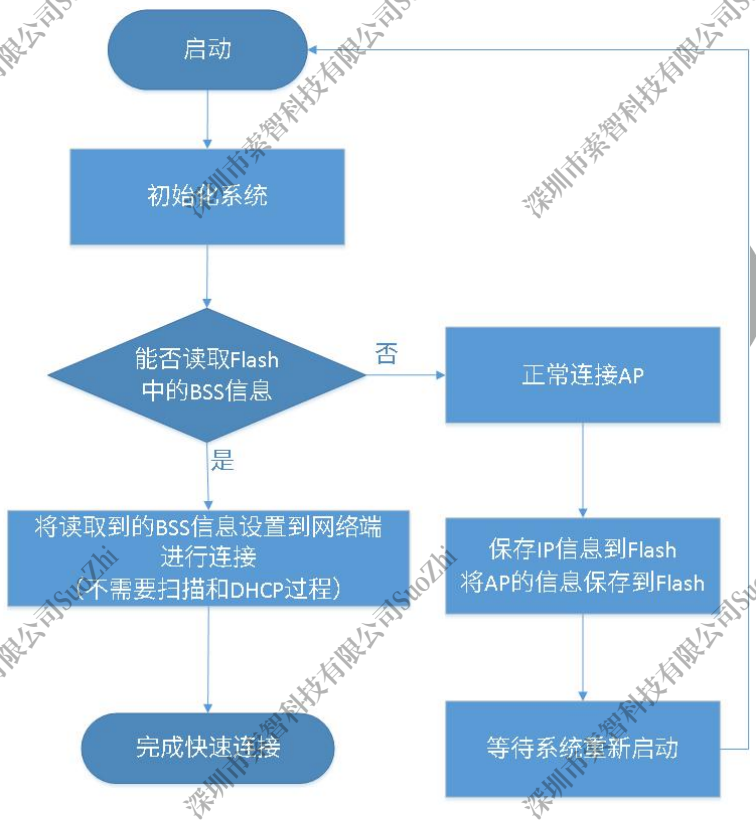
为了实现启动之后快速连接 AP，需要在第一次连接 AP 之后，将 AP 对应的信息保存到 Flash 中。在后续启动的时候可以从 Flash 中读取 AP 信息，省去了扫描和 DHCP 的过程。

因此，要完成快速连接的功能，必须有一次普通连接的过程。

3.1.1 实现流程

冷启动快连的大致实现流程如下：

图 3-1 冷启动快连实现流程



3.1.2 保存信息

为了能快速获取需要连接的 AP 信息，需要将这些信息按照固定格式保存在 Flash 指定的位置。其中部分信息是在 sysinfo 中使用的，所以可以直接保存在该区域中。剩余的信息需要用户在 Flash 上自定义一块区域进行保存。

具体需要保存的信息以及存放位置如下表所示：

表 3-1 WLAN 冷启动快连需要保存的信息

需要保存的内容	内容说明	保存位置
是否使用 DHCP 的标志	用于选择使用 DHCP 获取 IP 地址还是使用静态 IP 地址	sysinfo 结构体

版权所有©广州芯之联科技有限公司。保留一切权利

需要保存的内容	内容说明	保存位置
静态 IP 地址	使用静态 IP 连接时的 IP 信息	sysinfo 结构体
SSID	需要连接的 SSID 字符串	自定义
十六进制 PSK 值	使用需要连接的 password（ASC-II 格式）计算得到的十六进制 PSK 值，使用该值进行连接不需要重新计算	自定义
BSS 信息	需要连接的 AP 的 BSS 信息，将其设置到网络端后可以省去扫描操作	自定义

上述表格中需要用户自定义保存位置的内容建议将其合并为一个结构体后，使用 FDCM 格式保存在 Flash 的指定位置。FDCM 格式具有可以多次写入后才进行擦除的优势，具体使用方式可以参考《XR806_FDCM 中间件_开发指南》。

3.2 启动时间

3.2.1 启动时间记录

在 XR806 SDK 上，获取时间使用的函数为 HAL_RTC_GetFreeRunTime()，然后使用接口 save_time()将时间信息保存在一块指定的内存区域中，最后使用接口 get_time()打印输出统计的各个时间点，以及它们之间的差值。

具体接口不在此处进行介绍，用户可以直接参考示例工程中的使用。

3.2.2 影响因素

由于整个启动流程与系统相关，所以系统中有多个影响启动时间的因素。其中主要的几个影响因素列举如下：

3.2.2.1 串口打印

串口打印会占用较多时间，关闭多余打印，可以有效地缩短时间。

XR806 SDK 中使用接口 stdout_enable()来动态开关打印，详细请参考代码中的调用。注意，使用此接口只会关闭硬件串口的输出，代码的运行仍然存在，所以还是会有小部分时间消耗在代码运行上。

3.2.2.2 CPU频率

提高 CPU 的频率相当于加快代码的运行速度，可以在一定程度上缩短启动时间。

3.2.2.3 Flash ctrl时钟

系统启动时需要将代码从 Flash 加载到 SRAM 中运行，因此提高 Flash ctrl 的时钟也可以缩短一些启动时间。如果大部分应用代码都放在 XIP 的话，该措施不会有明显的效果。

版权所有©广州芯之联科技有限公司。保留一切权利

4 应用说明

4.1 应用简述

WLAN 冷启动快连方案已经内嵌到 XR806 SDK 中，用户通过 WLAN 冷启动快连方案提供的接口即可调用相关功能。

4.2 配置说明

WLAN 冷启动快连方案的启用需要使能 SDK 的 NET 和 WLAN 相关功能，具体配置项如下表所示。

表 4-1 XR806 WLAN 模块工程配置说明

配置项	配置说明
WLAN 功能使能	设置说明： 此项配置用于在 SDK 中启用 WLAN 功能。 设置方式： 1、在控制台执行命令“make menuconfig”，将“NET and WLAN”功能项打开。 2、在 prj_config.h 文件，添加或修改以下定义，其中 1 为使能 NET 功能，0 为不使能 NET 功能。 /* network and wlan enable/disable */ #define PRJCONF_NET_EN 1

4.3 接口说明

WLAN 冷启动快连方案总共提供 3 套接口，主要功能点包括 WLAN 连接操作、sysinfo 操作和 WLAN BSS 信息操作等。

表 4-2 WLAN 冷启动快连接口简介

接口名	简要介绍
WLAN 连接操作接口	用于配置连接指定的 AP
sysinfo 操作接口	用于读写 sysinfo 中的 IP 信息
WLAN BSS 信息操作接口	用于获取和设置 AP 的 BSS 信息

版权所有©广州芯之联科技有限公司。保留一切权利

4.3.1 WLAN 连接操作接口

4.3.1.1 wlan_sta_set

表 4-3 wlan_sta_set 接口函数说明

信息项	说明
原型	int wlan_sta_set (uint8_t *ssid, uint8_t ssid_len, uint8_t *psk);
功能	STA 模式下设置需要连接的 AP 的 ssid 和 psk
参数	uint8_t *ssid 含义解释：设置需要连接的 ssid 字符串 uint8_t ssid_len 含义解释：ssid 字符串的长度 uint8_t *psk 含义解释：设置需要连接的 password，需要符合以下任意一种格式要求 1. AP 是 open 模式时，该参数设置为 null 或者空字符串指针； 2. AP 是加密模式时，使用 ASC-II 格式字符串，长度为 8~63； 3. AP 是加密模式时，使用十六进制格式字符串，长度为 64。
返回值	0：成功 -1：失败

注意

在调用此函数后，若参数 psk 是 ASC-II 格式字符串，网络端会根据 SSID 和 psk 计算出一个 32-byte 的十六进制 PSK 值，需要大约 1s 的时间。

所以在快连应用中，应该在第一次连接的时候调用 wlan_sta_gen_psk()函数计算出该十六进制 PSK 值，然后保存起来，后续快连模式时直接设置该十六进制 PSK 值，就不需要 1s 的计算时间了。

4.3.1.2 wlan_sta_enable

表 4-4 wlan_sta_enable 接口函数说明

信息项	说明
原型	int wlan_sta_enable (void);
功能	STA 模式使能，Wi-Fi 模块会自动进行扫描和连接的操作
参数	无
返回值	0：成功 -1：失败

版权所有©广州芯之联科技有限公司。保留一切权利

4.3.1.3 wlan_sta_gen_psk

表 4-5 wlan_sta_gen_psk 接口函数说明

信息项	说明
原型	int wlan_sta_gen_psk(wlan_gen_psk_param_t *param);
功能	用于生成 32-byte 的十六进制 PSK 值，在调用 wlan_sta_set()函数时，将此十六进制 PSK 值作为参数传入的话，网络端不会再进行计算，而是直接使用该十六进制 PSK 值。 需要注意的是，wlan_sta_set()函数的 psk 参数格式为 64-byte 长度的十六进制格式的字符串，因此需要将 wlan_sta_gen_psk()函数生成的 32-byte 十六进制 PSK 数组转化为字符串形式。
参数	wlan_gen_psk_param_t *param 含义解释：包含了用于生成十六进制 PSK 值信息的结构体指针，具体定义参考表 4-6
返回值	0：成功 -1：失败

表 4-6 wlan_gen_psk_param_t 结构体的成员说明

成员项	说明
uint8_t ssid[WLAN_SSID_MAX_LEN]	含义解释：需要连接的 AP 的 SSID 字符串 使用说明：最大字符串长度为 32 byte
uint8_t ssid_len	含义解释：需要连接的 AP 的 ssid 字符串长度
char passphrase[WLAN_PASSPHRASE_MAX_LEN + 1]	含义解释：需要连接的 AP 的 password 字符串 使用说明：最大字符串长度为 63
uint8_t psk[WLAN_PSK_HEX_LEN]	含义解释：生成的十六进制 PSK 值 使用说明：长度为 32 byte

4.3.2 sysinfo 操作接口

4.3.2.1 sysinfo_get

表 4-7 sysinfo_get 接口函数说明

信息项	说明
原型	struct sysinfo *sysinfo_get(void)
功能	获取当前系统使用的 sysinfo 结构体指针，具体结构体中会使用到的成员定义参考表 4-8
参数	无

版权所有©广州芯之联科技有限公司。保留一切权利

信息项	说明
返回值	系统当前使用的 sysinfo 结构体指针，若执行失败返回 null

表 4-8 sysinfo 结构体的成员说明

成员项	说明
uint32_t sta_use_dhcp	含义解释：STA 模式连接时是否启用 DHCP 获取 IP 地址 使用说明：设置为 1 表示启用 DHCP 获取 IP 地址，设置为 0 表示使用静态 IP 地址，此时 IP 地址定义在成员 netif_sta_param 中
enum wlan_mode wlan_mode	含义解释：系统默认的 WLAN 启动模式 使用说明：取值为{WLAN_MODE_STA | WLAN_MODE_HOSTAP | WLAN_MODE_MONITOR}
struct sysinfo_netif_param netif_sta_param	含义解释：STA 模式使用静态 IP 进行连接时的 IP 信息 使用说明：具体定义参考表 4-9，在 sta_use_dhcp 置为 1 时，此参数不会被使用

注意

sysinfo 是应用层定义的结构，用户可自行添加和删除成员。上述表格仅列举冷启动快连模块会使用的相关成员。如需确认完整定义，可直接参考代码实现。

表 4-9 sysinfo_netif_param 结构体的成员说明

成员项	说明
ip4_addr_t ip_addr	含义解释：设备使用的 IP 地址
ip4_addr_t net_mask	含义解释：设备使用的 mask
ip4_addr_t gateway	含义解释：设备使用的网关地址

4.3.2.2 sysinfo_save

表 4-10 sysinfo_save 接口函数说明

信息项	说明
原型	int sysinfo_save(void)
功能	将当前使用的 sysinfo 数据保存到 flash 中，以便下次系统启动时使用
参数	无
返回值	0：成功

版权所有©广州芯之联科技有限公司。保留一切权利

信息项	说明
	-1：失败

4.3.3 WLAN BSS 信息操作接口

4.3.3.1 wlan_sta_get_bss_size

表 4-11 wlan_sta_get_bss_size 接口函数说明

信息项	说明
原型	int wlan_sta_get_bss_size(uint32_t * size);
功能	获取当前连接的 AP 的 BSS 信息数据大小
参数	uint32_t * size 含义解释：返回当前连接的 AP 的 BSS 信息数据大小
返回值	0：成功 -1：失败

4.3.3.2 wlan_sta_get_bss

表 4-12 wlan_sta_get_bss 接口函数说明

信息项	说明
原型	int wlan_sta_get_bss(wlan_sta_bss_info_t * bss_get);
功能	获取当前连接的 AP 的 BSS 信息数据
参数	wlan_sta_bss_info_t * bss_get 含义解释：返回当前连接的 AP 的 BSS 信息结构体指针，具体定义参考表 4-13 使用说明：需要调用 wlan_sta_get_bss_size()来获取 BSS 数据的大小，然后动态分配相应大小的内存块并初始化为 0，将该内存块的指针赋给 wlan_sta_bss_info_t 的 bss 成员即可。使用完毕后需要释放该内存块
返回值	0：成功 -1：失败

表 4-13 wlan_sta_bss_info_t 结构体的成员说明

成员项	说明
uint8_t *bss	含义解释：实际网络端使用的 BSS 数据指针 使用说明:需要动态分配 BSS 数据大小的内存块并初始化为 0，传入 wlan_sta_get_bss()函数后会填充相应的 BSS 数据至该内存块
uint32_t size	含义解释：BSS 数据的大小

版权所有©广州芯之联科技有限公司。保留一切权利

成员项	说明
	使用说明：需要调用 wlan_sta_get_bss_size()来获取

4.3.3.3 wlan_sta_set_bss

表 4-14 wlan_sta_set_bss 接口函数说明

信息项	说明
原型	int wlan_sta_set_bss(wlan_sta_bss_info_t * bss_set);
功能	设置当前需要连接的 AP 的 BSS 信息到网络端
参数	wlan_sta_bss_info_t * bss_set 含义解释：设置当前需要连接的 AP 的 BSS 信息结构体指针
返回值	0：成功 -1：失败

4.4 指令说明

基于上述接口功能，XR806 上提供了一些指令方便用户对相关功能进行操作和调试。

4.4.1 连接配置指令

信息项	说明
原型	net fc config <ssid> <password>
功能	设置需要连接的 SSID 和 password，并将其保存下来
参数	<ssid>：需要连接的 SSID 字符串 <password>：需要连接的 password，如果是不加密的 AP，则不需要此参数
响应	OK
示例	设置 SSID 和 password 分别为 your_ssid 和 your_password： net fc config your_ssid your_password
备注	无

4.4.2 连接使能指令

信息项	说明
原型	net fc enable
功能	使能 STA 模式，连接指定的 AP
参数	无
响应	OK

版权所有©广州芯之联科技有限公司。保留一切权利

信息项	说明
示例	net fc enable
备注	无

4.4.3 清除保存的 BSS 信息指令

信息项	说明
原型	net fc clear_bss
功能	清除保存在 Flash 上的 BSS 信息，设置连接 AP 时使用 DHCP 方式获取 IP 地址，下次启动连接时会采用普通连接
参数	无
响应	OK
示例	net fc clear_bss
备注	无

深圳市索智科技有限公司SuoZhi

版权所有©广州芯之联科技有限公司。保留一切权利

5 示例说明

5.1 示例简介

本示例通过演示 WLAN 的 STA 模式进行普通连接指定的路由器之后重启进行快连的过程，简要介绍对应接口的使用方法。

5.1.1 获取方式

WLAN 冷启动快连方案示例有示例工程代码，位于 XR806 SDK 的/project/example/wlan_csfc 目录，以下此示例工程简称为 wlan_csfc 示例工程。

说明

XR806 SDK 可在以下 GitHub 仓库获取：https://github.com/XradioTech/xr806_sdk.git

5.1.2 准备工作

wlan_csfc 示例工程的硬件准备有如下。

1. 评估板：运行示例工程代码。
2. 串口线：连接评估板的 Uart0 插针，用于 console 控制台的输入输出。
3. PC 机：用于镜像烧录和 console 控制的输入输出。

wlan_csfc 示例工程的软件准备，包括烧写工具、代码编译和烧写操作，请参见《XR806_SDK_快速入门指南》。

5.1.3 操作步骤

wlan_csfc 示例工程操作步骤如下：

1. 按下 reset 键，系统启动；
2. 系统初始化，然后尝试从 Flash 指定位置读取 BSS 信息，此时会读取失败，进入普通连接模式；
3. 输入命令连接指定的 AP，以下以 SSID 为“your_ssid”、password 为“your_password”为例：

```
net fc config your_ssid your_password
net fc enable
```

4. 等待连接上 AP 之后，IP 信息和 BSS 信息会自动保存到 flash 的指定位置，然后会打印本次连接耗时；
5. 再次按下 reset 键，重启系统；
6. 系统初始化，然后会读取到保存好的 BSS 信息，并将其设置到网络端；
7. 进入快连模式，此时进行连接不会有扫描和 DHCP 过程，连接完成后会打印本次连接耗时；
8. 如果用户想连接其它 AP，则输入以下命令清除 Flash 上保存的 AP 信息，再从步骤 1 开始操作即可：

```
net fc clear_bss
```

版权所有©广州芯之联科技有限公司。保留一切权利

5.2 关键实现

系统平台的初始化在 XR806 SDK 的 platform_init()函数中完成，具体请阅读此函数。之后的代码流程与上一章节的操作步骤相同。

具体操作见下面的示例代码。

```
void connect_ap(void)
{
    printf("Wait for link up...\n");
    printf("use this cmd to connect your ap with your own ssid and password:\n"
            "\t\"net fc config your_ssid your_password\"\n\t\"net fc enable\"\n");
    while (!g_ap_connected) {
        OS_MSleep(10);
    }
    save_time((uint32_t)HAL_RTC_GetFreeRunTime(), 2);
    printf("Connect AP success!\n");
    struct sysinfo *sysinfo = sysinfo_get();
    if (sysinfo == NULL) {
        printf("sysinfo %p\n", sysinfo);
        return;
    }

    // Save IP info to flash
    sysinfo->sta_use_dhcp = 0;
    memcpy(&sysinfo->netif_sta_param.ip_addr, &g_wlan_netif->ip_addr, sizeof(ip_addr_t));
    memcpy(&sysinfo->netif_sta_param.gateway, &g_wlan_netif->gw, sizeof(ip_addr_t));
    memcpy(&sysinfo->netif_sta_param.net_mask, &g_wlan_netif->netmask, sizeof(ip_addr_t));
    sysinfo_save();
}

int save_bss_to_flash(bss_info_t * pbss_info)
{
    int ret = 0;
    uint32_t size;
    wlan_sta_bss_info_t bss_get;
    fdcm_handle_t * bss_fdcm_hdl;

    if (!g_ap_connected) {
        printf("Please connect AP first!\n");
        ret = -1;
        return ret;
    }
    // Try to get current bss info size
    ret = wlan_sta_get_bss_size(&size);
```

版权所有©广州芯之联科技有限公司。保留一切权利

```
if (ret != 0) {
    printf("Get current bss info size failed!\n");
    return ret;
}
bss_get.size = size;
bss_get.bss = malloc(size);

// Try to get current bss info
ret = wlan_sta_get_bss(&bss_get);
if (ret != 0) {
    printf("Get current bss info failed!\n");
    return ret;
}

// Gererate 32 bytes HEX psk, so that we don't need to calcute next time
wlan_gen_psk_param_t param;
if (strlen(sta_psk) == 64) {
    hex2bin(param.psk, sta_psk, 32);
} else {
    param.ssid_len = strlen(sta_ssid);
    memcpy(param.ssid, sta_ssid, param.ssid_len);
    strlcpy(param.passphrase, sta_psk, sizeof(param.passphrase));
    ret = wlan_sta_gen_psk(&param);
    if (ret != 0) {
        printf("fail to generate psk\n");
        ret = -1;
        return ret;
    }
}

// Save current bss info to flash
memset(pbss_info, 0, sizeof(bss_info_t));
memcpy(pbss_info->ssid, sta_ssid, strlen(sta_ssid));
memcpy(pbss_info->psk, param.psk, 32);
pbss_info->bss_size = size;
memcpy(pbss_info->bss, bss_get.bss, size);
bss_fdcm_hdl = fdcm_open(BSS_FLASH_NUM, BSS_FLASH_ADDR, BSS_FLASH_SIZE);
if (bss_fdcm_hdl == NULL) {
    printf("fdcm open failed, hdl %p\n", bss_fdcm_hdl);
    ret = -1;
    return ret;
}
fdcm_write(bss_fdcm_hdl, pbss_info, sizeof(bss_info_t));
fdcm_close(bss_fdcm_hdl);
```

版权所有©广州芯之联科技有限公司。保留一切权利

```
    free(bss_get.bss);
    printf("Save bss info done!\n");
    return ret;
}

void connect_ap_normal(void)
{
    connect_ap();

    // Save new bss info to flash
    save_bss_to_flash(&g_bss_info);
}

void connect_ap_fast(bss_info_t * pbss_info)
{
    char psk_buf[64];
    char *p;
    int i;
    p = psk_buf;

    FC_DEBUG("Set old bss info!\n");
    for (i = 0;i < 32;i++) {
        sprintf(p, "%02x", pbss_info->psk[i]);
        p += 2;
    }
    wlan_sta_set((uint8_t *)pbss_info->ssid, strlen((char *)pbss_info->ssid), (uint8_t *)psk_buf);

    FC_DEBUG("Try to connect AP\n");
    wlan_sta_enable();

    FC_DEBUG("Wait for link up...\n");
    while (!g_ap_connected) {
        OS_MSleep(10);
    }
    save_time((uint32_t)HAL_RTC_GetFreeRunTime(), 2);
    // Fast connect AP success
}

int get_bss_from_flash(bss_info_t * pbss_info)
{
    int ret;
    uint32_t size;
    wlan_sta_bss_info_t bss_set;
    fdcm_handle_t * bss_fdcm_hdl;
```

深圳市索智科技有限公司SuoZhi

```
    bss_fdcm_hdl = fdcm_open(BSS_FLASH_NUM, BSS_FLASH_ADDR, BSS_FLASH_SIZE);
    if (bss_fdcm_hdl == NULL) {
        printf("fdcm open failed, hdl %p\n", bss_fdcm_hdl);
        ret = -1;
        return ret;
    }
    size = fdcm_read(bss_fdcm_hdl, pbss_info, sizeof(bss_info_t));
    fdcm_close(bss_fdcm_hdl);
    if (size != sizeof(bss_info_t)) {
        printf("fdcm read failed, size %d\n", size);
        ret = -1;
        return ret;
    }
    if (pbss_info->bss_size == 0) {
        printf("empty bss info\n");
        ret = -1;
        return ret;
    }

    FC_DEBUG("SSID:%s\n", pbss_info->ssid);
    FC_DEBUG("PSK:");
    for (int i = 1; i < 33; ++i)
        FC_DEBUG("%02x", pbss_info->psk[i-1]);
    FC_DEBUG("\n");
    FC_DEBUG("BSS size:%d\n", pbss_info->bss_size);

    FC_DEBUG("Set current bss info!\n");
    bss_set.size = pbss_info->bss_size;
    bss_set.bss = malloc(bss_set.size);
    memcpy(bss_set.bss, pbss_info->bss, bss_set.size);
    ret = wlan_sta_set_bss(&bss_set);
    free(bss_set.bss);
    return ret;
}

void fast_connect_example(void)
{
    FC_DEBUG("Init wlan message receiver\n");
    net_msg_rcv_init();

    FC_DEBUG("Begin fast connect example\n");
    FC_DEBUG("Try to get old bss info in flash...\n");
    if (get_bss_from_flash(&g_bss_info)) {
```

深圳市索智科技有限公司SuoZhi

```
#if !FC_DEBUG_EN
        stdout_enable(1);
#endif
        printf("Get old bss failed!\n");
        printf("Begin normal connection\n");
        connect_ap_normal();
        printf("The first connection is complete, please reboot to run fast connection!\n");
    } else {
        FC_DEBUG("Get old bss info success!\n");
        FC_DEBUG("Begin fast connection\n");
        connect_ap_fast(&g_bss_info);
#if !FC_DEBUG_EN
        stdout_enable(1);
#endif
    }
}

int main(void)
{
    uint32_t time_eob, time_eop;
    time_eob = (uint32_t)HAL_RTC_GetFreeRunTime();
#if !FC_DEBUG_EN
    stdout_enable(0);
#endif
    platform_init();
    time_eop = (uint32_t)HAL_RTC_GetFreeRunTime();
    save_time(time_eob, 0);
    save_time(time_eop, 1);
    fast_connect_example();
    get_time();
    return 0;
}
```

5.3 效果展示

在评估板中运行 wlan_csfc 示例程序后，第一次普通连接时控制台中的打印如下所示：

```
Get old bss failed!
Begin normal connection
Wait for link up...
use this cmd to connect your ap with your own ssid and password:
        "net fc config your_ssid your_password"
```

深圳市索智科技有限公司SuoZhi

```
            "net fc enable"
net fc config ssid password
<ACK> 200 OK
net fc enable
$ <ACK> 200 OK
$ en1: Trying to associate with 80:37:73:f7:dc:f4 (SSID='ssid' freq=2412 MHz)
[net INF] msg <wlan scan success>
en1: Associated with 80:37:73:f7:dc:f4
en1: WPA: Key negotiation completed with 80:37:73:f7:dc:f4 [PTK=CCMP GTK=CCMP]
en1: CTRL-EVENT-CONNECTED - Connection to 80:37:73:f7:dc:f4 completed [id=0 id_str=]
[net INF] msg <wlan connected>
[net INF] netif is link up
[net INF] start DHCP...
[net INF] netif (IPv4) is up
[net INF] address: 192.168.51.5
[net INF] gateway: 192.168.51.1
[net INF] netmask: 255.255.255.0
[net INF] msg <network up>
Connect AP success!
Save bss info done!
The first connection is complete, please reboot to run fast connection!

begin app:          028687us            delta:28687us
end pf init:        118343us            delta:89656us
end connection: 8073062us           delta:7954719us
delta of begin app to end connection:     8044375us
```

再次按下 reset 键进行快连时控制台的打印如下所示：

```
begin app:          028687us            delta:28687us
end pf init:        118343us            delta:89656us
end connection: 252937us            delta:134594us
delta of begin app to end connection:     224250us
```

对于以上统计时间说明如下：

1. begin app：进入 app 的时间点
2. end pf init：平台初始化完成的时间点
3. end connection：完成 AP 连接，获取到 IP 的时间点
4. delta of begin app to end connection：从进入到 app 的时间到完成 AP 连接的时间差

注意

完成 AP 连接的时间点与网络环境和 AP 行为有关，可能会有较大的偏差.

版权所有©广州芯之联科技有限公司。保留一切权利

著作权声明

版权所有©2020 广州芯之联科技有限公司。保留一切权利。

本文档及内容受著作权法保护，其著作权由广州芯之联科技有限公司（“芯之联”）拥有并保留一切权利。

本文档是芯之联的原创作品和版权财产，未经芯之联书面许可，任何单位和个人不得擅自摘抄、复制、修改、发表或传播本文档内容的部分或全部，且不得以任何形式传播。

商标声明

、 （不完全列举）均为广州芯之联科技有限公司的商标或者注册商标。在本文档描述的产品中出现的其它商标，产品名称，和服务名称，均由其各自所有人拥有。

免责声明

您购买的产品、服务或特性应受您与广州芯之联科技有限公司（“芯之联”）之间签署的商业合同和条款的约束。本文档中描述的全部或部分产品、服务或特性可能不在您所购买或使用的范围内。使用前请认真阅读合同条款和相关说明，并严格遵循本文档的使用说明。您将自行承担任何不当使用行为（包括但不限于如超压，超频，超温使用）造成的不利后果，芯之联概不负责。

本文档作为使用指导仅供参考。由于产品版本升级或其他原因，本文档内容有可能修改，如有变更，恕不另行通知。芯之联尽全力在本文档中提供准确的信息，但并不确保内容完全没有错误，因使用本文档而发生损害（包括但不限于间接的、偶然的、特殊的损失）或发生侵犯第三方权利事件，芯之联概不负责。本文档中的所有陈述、信息和建议并不构成任何明示或暗示的保证或承诺。

本文档未以明示或暗示或其他方式授予芯之联的任何专利或知识产权。在您实施方案或使用产品的过程中，可能需要获得第三方的权利许可。请您自行向第三方权利人获取相关的许可。芯之联不承担也不代为支付任何关于获取第三方许可的许可费或版税（专利税）。芯之联不对您所使用的第三方许可技术做出任何保证、赔偿或承担其他义务。