

# ASR582X系列 WLAN 开发指南

文档版本 1.0.1

发布日期 2023-09-26

版权所有 © 2023 翱捷科技

### 关于本文档

本文档旨在介绍 ASR582X 系列芯片 Wi-Fi 功能的相关 API 接口的使用方法、API 用例以及注意事项。

#### 读者对象

本文档主要适用于以下工程师:

- 软件工程师
- 技术支持工程师

### 产品型号

本文档适用于 ASR582X 系列 Wi-Fi+BLE Combo SoC 芯片

#### 版权公告

版权归 © 2023 翱捷科技股份有限公司所有。保留一切权利。未经翱捷科技股份有限公司的书面许可,不得以任何形式或手段复制、传播、转录、存储或翻译本文档的部分或所有内容。

### 商标声明

△S⊋ ASR、翱捷和其他翱捷商标均为翱捷科技股份有限公司的商标。

本文档提及的其他所有商标名称、商标和注册商标均属其各自所有人的财产,特此声明。

### 免责声明

翱捷科技股份有限公司对本文档内容不做任何形式的保证,并会对本文档内容或本文中介绍的产品进行不定期更新。

本文档仅作为使用指导,本文的所有内容不构成任何形式的担保。本文档中的信息如有变更,恕 不另行通知。

本文档不负任何责任,包括使用本文档中的信息所产生的侵犯任何专有权行为的责任。

### 翱捷科技股份有限公司

地址: 上海市浦东新区科苑路399号张江创新园10号楼9楼 邮编: 201203

官网: http://www.asrmicro.com/

### 文档修订历史

日期	版本号	发布说明	
2022.10.08	0.0.1	首次发布。	* C
2022.10.14	0.0.2	更新参数注释。	7/0
2022.12.13	1.0.0	更新文档格式和布局。	
2023.09.26	1.0.1	更新节 1.1.8 和节 1.1.9。	

1.	应用	接口 API	1
	1.1	Wi-Fi 接口	1
		1.1.1 lega_wlan_init	1
		1.1.2 lega_wlan_deinit	1
		1.1.3 lega_wlan_open	
		1.1.4 lega_wlan_close	2
		1.1.5 lega_wlan_start_scan	2
		1.1.6 lega_wlan_start_scan_detail	2
		1.1.7 lega_wlan_start_scan_active	2
		1.1.8 lega_wlan_get_mac_address	3
		1.1.9 lega_wlan_set_mac_address	3
		1.1.10 lega_wlan_get_ip_status	3
		1.1.11 lega_wlan_get_link_status	
		1.1.12 lega_wlan_get_associated_apinfo	
		1.1.13 lega_wlan_start_monitor	
		1.1.14 lega_wlan_stop_monitor	
		1.1.15 lega_wlan_monitor_set_channel	
		1.1.16 lega_wlan_get_channel	
		1.1.17 lega_wlan_set_ps_options	
		1.1.18 lega_wlan_set_ps_mode	
		1.1.19 lega_wlan_register_monitor_cb	
		1.1.20 lega_wlan_send_raw_frame	
		1.1.21 lega_wlan_start_debug_mode	
		1.1.22 lega_wlan_stop_debug_mode	
		1.1.23 lega_wlan_ip_got_cb_register	
		1.1.24 lega_wlan_stat_chg_cb_register	
		1.1.25 lega_wlan_scan_compeleted_cb_register	
		1.1.26 lega_wlan_associated_ap_cb_register	
		1.1.27 lega_wlan_ap_peer_change_cb_register	
		1.1.28 lega_drv_open_dcdc_pfm	
		1.1.29 lega_drv_close_dcdc_pfm	
		1.1.30 lega_drv_rco_cal	
		1.1.31 lega_wlan_set_country_code	
		1.1.32 lega_wlan_get_country_code	
		1.1.33 lega_get_client_ip_mac	
		1.1.34 lega_wlan_err_stat_cb_register	
		1.1.35 lega_wlan_get_wifi_mode	9

		1.1.36 lega_wlan_get_softap_info	9
		1.1.37 lega_wlan_softap_deauth_peer	. 10
	1.2	Wi-Fi 接口参数	. 11
		1.2.1 Wi-Fi 开启参数	. 11
		1.2.2 Wi-Fi 获取 IP 信息参数	. 12
		1.2.3 Wi-Fi 获取链路信息参数	. 12
		1.2.4 Wi-Fi 扫描完成回调函数参数	. 12
		1.2.5 Wi-Fi 关联完成获取 AP 信息回调函数参数	. 13
		1.2.6 Wi-Fi 运行状态信息	. 13
		1.2.7 Wi-Fi 运行错误状态信息	. 14
		1.2.8 SOFTAP 模式下获取对端信息参数	. 14
2.	Wi-Fi	接口 API 用例	. 15
	2.1	Sta 模式的开启与关闭	. 15
	2.2	Sniffer 模式的开启与关闭	. 15
	2.3	Softap 模式的开启与关闭	. 15
3.	_	<b>事项</b>	. 16



### 1.

# 应用接口 API

### 1.1 Wi-Fi 接口

Wi-Fi 接口头文件位于 lib\wifi\lega\_wlan\_api.h。

一类为供用户调用的 Wi-Fi 功能控制接口,一类为用户需要注册给 Wi-Fi 协议栈的回调函数注册接口。

### 1.1.1 lega\_wlan\_init

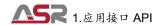
功能	Wi-Fi 协议栈软件初始化
函数定义	int lega_wlan_init(void)
参数	无
返回	0: 命令下达成功 非 0: 命令下达失败
注意	在 lega_wlan_open 之前调用(可在系统初始化的时候调用一次)

### 1.1.2 lega\_wlan\_deinit

功能	Wi-Fi 协议栈注销
函数定义	int lega_wlan_deinit(void)
参数	无
返回	0: 命令下达成功 非 0: 命令下达失败
注意	在 lega_wlan_close 之后调用(若系统不再使用 WLAN 功能,可以调用)

### 1.1.3 lega\_wlan\_open

	开启Wi-Fi(用于STATION模式和SOFTAP模式)
	用于开启 STATION 模式时需注意:
	通过入参配置开启 station 模式(必选),会开启 Wi-Fi 协议栈并工作在 station
	模式
T-I- Ab	通过入参配置开启 station 模式(必选)、配置欲连接 AP 的 SSID(必选)和
功能	密码(为加密 AP 时必选),Wi-Fi 协议栈会自动执行 Wi-Fi 连接过程直至获取
	到 IP 地址
	用于开启 SOFTAP 模式时需注意:
	通过入参配置开启 softap 模式(必选)、配置 softap 的 SSID(必选)、密码(必
	选,不加密时用0代替)和信道(可选)
函数定义	int lega_wlan_open(lega_wlan_init_type_t* init_info)
参数	lega_wlan_init_type_t * init_info
	结构体定义详见 1.2 章节
返回	0: 命令下达成功
	非 0: 命令下达失败



### 1.1.4 lega\_wlan\_close

功能	关闭 Wi-Fi(用于 STATION 模式和 SOFTAP 模式)	
函数定义	int lega_wlan_close (void)	
参数	无	
返回	0: 命令下达成功 非 0: 命令下达失败	

### 1.1.5 lega\_wlan\_start\_scan

功能	Wi-Fi 扫描(用于 STATION 模式)
函数定义	int lega_wlan_start_scan (void)
参数	无
返回	0: 命令下达成功 非 0: 命令下达失败
注意	开启 Wi-Fi STA 模式后调用此接口

### 1.1.6 lega\_wlan\_start\_scan\_detail

功能	Wi-Fi 特定扫描(用于 STATION 模式,携带参数指定扫描)		
函数定义	int lega_wlan_start_scan_detail(char *ssid, int channel, char *bssid)		
参数	<ul> <li>char *ssid 指定具体 ssid 的扫描</li> <li>int channel 指定在某个信道上扫描路由器</li> <li>char *bssid 指定具体 bssid 的扫描</li> </ul>		
返回	0: 命令下达成功 非 0: 命令下达失败		
注意	开启 Wi-Fi STA 模式后调用此接口		

## 1.1.7 lega\_wlan\_start\_scan\_active

功能	Wi-Fi 特定 ssid 扫描(用于 STATION 模式,携带参数指定扫描)		
函数定义	int lega_wlan_start_scan_active(const char *ssid, uint8_t is_scan_advance);		
参数	● char *ssid 指定具体 ssid 的扫描 ● uint8_t is_scan_advance 当前未使用		
返回	0: 命令下达成功 非 0: 命令下达失败		
注意	开启 Wi-Fi STA 模式后调用此接口		



### 1.1.8 lega\_wlan\_get\_mac\_address

功能	获取本机的 Wi-Fi MAC 地址		
函数定义	int lega_wlan_get_mac_address (uint8_t *mac_addr)		
参数	● uint8_t *mac_addr Wi-Fi 协议栈将 6 字节 MAC 地址填入 mac_addr 指向的内存空间		
返回	0: 命令下达成功 非 0: 命令下达失败		
注意	该原理是先从 efuse 中读取 MAC 地址,若 efuse 中没有,则再从 flash 中读取 MAC 地址		

### 1.1.9 lega\_wlan\_set\_mac\_address

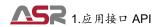
功能	调试阶段用于设置本机的 Wi-Fi MAC 地址到 flash 空间		
函数定义	int lega_wlan_set_mac_address (uint8_t *mac_addr)		
参数	● uint8_t *mac_addr 将 6 字节 MAC 地址填入 mac_addr 指向的 flash 空间		
返回	0: 命令下达成功 -1: 入参 mac_addr 为空指针 -2: efuse 中已有 MAC 地址,不能在 flash 中设置 MAC 地址 -3: MAC 地址是无效的		
注意	不建议使用,通常情况下,出厂时已将 MAC 地址固化到 efuse 中。		

### 1.1.10 lega\_wlan\_get\_ip\_status

功能	获取 IP 相关信息(用于 STATION 模式和 SOFTAP 模式)
函数定义	lega_wlan_ip_stat_t * lega_wlan_get_ip_status(void);
参数	无
返回	lega_wlan_ip_stat_t *, IP 相关信息指针,NULL 表示无法读取 结构体定义详见 1.2 章节

### 1.1.11 lega\_wlan\_get\_link\_status

功能	获取连接链路相关信息(用于 STATION 模式)
函数定义	int lega_wlan_get_link_status (lega_wlan_link_stat_t *link_status)
参数	● lega_wlan_link_stat_t *link_status Wi-Fi 协议栈将 Wi-Fi 连接链路相关信息填入 link_status 指向的内存空间 结构体定义详见 1.2 章节
返回	0: 命令下达成功 非 0: 命令下达失败



### 1.1.12 lega\_wlan\_get\_associated\_apinfo

功能	关联 AP 成功后获取 AP 相关信息(用于 STATION 模式)
函数定义	lega_wlan_ap_info_adv_t *lega_wlan_get_associated_apinfo(void)
参数	无
返回	lega_wlan_ap_info_adv_t *, AP 相关信息指针,NULL 表示无法读取 结构体定义详见 1.2 章节

### 1.1.13 lega\_wlan\_start\_monitor

功能	开启 Wi-Fi sniffer 模式(用于 SNIFFER 模式)
函数定义	int lega_wlan_start_monitor (void)
参数	无
返回	0: 命令下达成功 非 0: 命令下达失败

### 1.1.14 lega\_wlan\_stop\_monitor

功能	关闭 Wi-Fi sniffer 模式(用于 SNIFFER 模式)
函数定义	int lega_wlan_stop_monitor (void)
参数	无
返回	0: 命令下达成功 非 0: 命令下达失败

### 1.1.15 lega\_wlan\_monitor\_set\_channel

配置 Wi-Fi sniffer 模式工作的信道(用于 SNIFFER 模式)
开启 Wi-Fi sniffer 模式后调用此接口
int lega_wlan_monitor_set_channel (int channel)
int channel
channel 配置范围:1-13
0: 命令下达成功
非 0: 命令下达失败

### 1.1.16 lega\_wlan\_get\_channel

功能	获取 STA 或者 SOFTAP 模式工作信道
函数定义	int lega_wlan_get_channel(void)
参数	无
返回	1~14: 当前 Wi-Fi 的工作信道 0: 获取工作信道失败



### 1.1.17 lega\_wlan\_set\_ps\_options

功能	配置 Wi-Fi powersave 相关参数(用于 STA 模式) 开启 Wi-Fi STA 模式后调用此接口
函数定义	int lega_wlan_set_ps_options(uint8_t listen_bc_mc, uint16_t listen_interval)
参数	<ul> <li>uint8_t listen_bc_mc 是否需要在睡眠前等待 AP 缓存的组播或广播报文标记</li> <li>uint16_t listen_interval 侦听 AP 发送 beacon 的间隔(一般范围 1 到 50)</li> </ul>
返回	0: 命令下达成功 非 0: 命令下达失败

### 1.1.18 lega\_wlan\_set\_ps\_mode

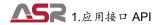
功能	配置 Wi-Fi powersave 相关参数(用于 STA 模式) 开启 Wi-Fi STA 模式并获取到 IP 地址后可调用此接口
函数定义	int lega_wlan_set_ps_mode(uint8_t ps_on)
参数	● uint8_t ps_on 是否支持 powersave 模式
返回	0: 命令下达成功 非 0: 命令下达失败

### 1.1.19 lega\_wlan\_register\_monitor\_cb

功能	注册 sniffer 模式下收包的回调函数(用于 SNIFFER 模式) 开启 Wi-Fi sniffer 模式前调用此接口
函数定义	int lega_wlan_register_monitor_cb (monitor_data_cb_t fn)
参数	● monitor_cb_t fn Wi-Fi 协议栈在 sniffer 模式下收到包后会调用 fn 将 MPDU 上报给用户 回调函数类型定义如下: typedef void (*monitor_cb_t)(uint8_t *data, int len, int rssi) Wi-Fi 协议栈会将收到的 MPDU 头地址通过 data 传入 Wi-Fi 协议栈会将收到的 MPDU 长度通过 len 传入 Wi-Fi 协议栈会将收到数据包的信号强度通过 rssi 传入
返回	0: 命令下达成功 非 0: 命令下达失败

### 1.1.20 lega\_wlan\_send\_raw\_frame

功能	用户构造 MPDU 发送接口(用于 STATION 模式和 SNIFFER 模式) 调用此接口发送用户层构造的 MPDU
函数定义	int lega_wlan_send_raw_frame (uint8_t *buf, int len)
参数	● uint8_t *buf 指向欲发送的 MPDU 的指针 ● int len MPDU 长度
返回	0: 命令下达成功 非 0: 命令下达失败



### 1.1.21 lega\_wlan\_start\_debug\_mode

功能	开启Wi-Fi 协议栈日志
	通过 uart 输出
函数定义	int lega_wlan_start_debug_mode (void)
参数	无
返回	0: 命令下达成功
	非 0: 命令下达失败

### 1.1.22 lega\_wlan\_stop\_debug\_mode

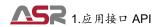
功能	关闭 Wi-Fi 协议栈日志	
2) HC	通过 uart 输出	
函数定义	int lega_wlan_stop_debug_mode (void)	
参数	无	
) F (F)	0: 命令下达成功	
返回	非 0: 命令下达失败	

### 1.1.23 lega\_wlan\_ip\_got\_cb\_register

功能	注册获取 IP 回调函数(用于 STATION 模式)	
函数定义	int lega_wlan_ip_got_cb_register (lega_wlan_cb_ip_got fn)	
参数	● lega_wlan_cb_ip_got fn Wi-Fi 协议栈在获取 IP 地址后调用此回调函数 回调函数类型定义如下: typedef void (*lega_wlan_cb_ip_got)( lega_wlan_ip_stat_t *ip_status) lega_wlan_ip_stat_t 结构体定义详见 1.2 章节	
返回	0: 命令下达成功 非 0: 命令下达失败	

### 1.1.24 lega\_wlan\_stat\_chg\_cb\_register

	注册 Wi-Fi 状态改变回调函数(用于 STATION 模式和 SOFTAP 模式) 对于 STATION 模式:	
	a. 如果连接非加密 AP,Wi-Fi 协议栈会在关联成功后调用回调函数	
功能	b. 如果连接加密 AP, Wi-Fi 协议栈会在 4 次握手成功后调用回调函数	
り用じ	c. Wi-Fi 协议栈会在 STATION 断开连接后调用回调函数	
	対于 SOFTAP 模式:	
	a. Wi-Fi 协议栈会在 SOFTAP 模式正常开启后调用回调函数	
	b. Wi-Fi 协议栈会在 SOFTAP 模式正常关闭后调用回调函数	
函数定义	int lega_wlan_stat_chg_cb_register (lega_wlan_cb_stat_chg fn)	
	lega_wlan_cb_stat_chg fn	
	Wi-Fi 协议栈在 Wi-Fi 关键状态改变后调用此回调函数	
参数	回调函数类型定义如下:	
	typedef void (*lega_wlan_cb_stat_chg)( lega_wifi_event_e wlan_event)	
	lega_wifi_event_e 结构体定义详见 1.2 章节	
No.	0: 命令下达成功	
返回	非 0: 命令下达失败	



### 1.1.25 lega\_wlan\_scan\_compeleted\_cb\_register

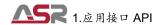
功能	注册 Wi-Fi 扫描完成回调函数(用于 STATION 模式)
函数定义	int lega_wlan_scan_compeleted_cb_register (lega_wlan_cb_scan_compeleted
	fn)
	lega_wlan_cb_scan_compeleted fn
	Wi-Fi 协议栈在扫描完成后调用此回调函数
参数	回调函数类型定义如下:
少奴	typedef void (*lega_wlan_cb_scan_compeleted)( lega_wlan_scan_result_t
	*result)
	lega_wlan_scan_result_t 结构体定义详见 1.2 章节
返回	0: 命令下达成功
	非 0: 命令下达失败
注意	最多可以扫描显示 32 个 ap,按照 rssi 的强弱由强到弱排序

### 1.1.26 lega\_wlan\_associated\_ap\_cb\_register

功能	注册 Wi-Fi 关联完成获取 AP 信息回调函数 (用于 STATION 模式)	
函数定义	int lega_wlan_associated_ap_cb_register(lega_wlan_cb_associated_ap fn)	
参数	● lega_wlan_cb_associated_ap fn Wi-Fi 协议栈在关联 AP 成功后调用此回调函数 回调函数类型定义如下: typedef void (*lega_wlan_cb_associated_ap)(lega_wlan_ap_info_adv_t) lega_wlan_ap_info_adv_t 结构体定义详见 1.2 章节	
返回	0: 命令下达成功 非 0: 命令下达失败	

### 1.1.27 lega\_wlan\_ap\_peer\_change\_cb\_register

功能	注册连接的客户端状态改变通知回调函数(用于 SOFTAP 模式)	
函数定义	int lega_wlan_ap_peer_change_cb_register(lega_wlan_cb_ap_peer_change	
	fn)	
	lega_wlan_cb_ap_peer_change fn	
	Wi-Fi 协议栈在热点的客户端连接成功或者断开调用此回调函数	
<b>全</b> 数	回调函数类型定义如下:	
参数	typedef void (*lega_wlan_cb_ap_peer_change)(lega_wlan_client_addr_info_t	
	*peer_info, uint8_t is_connect)	
	lega_wlan_client_addr_info_t 结构体定义详见 1.2 章节	
返回	0: 命令下达成功	
	非 0: 命令下达失败	



### 1.1.28 lega\_drv\_open\_dcdc\_pfm

功能	打开 DCDC 的 PFM 模式
函数定义	void lega_drv_open_dcdc_pfm(void)
参数	无
返回	无

### 1.1.29 lega\_drv\_close\_dcdc\_pfm

功能	关闭 DCDC 的 PFM 模式
函数定义	void lega_drv_close_dcdc_pfm(void)
参数	无
返回	无

#### 1.1.30 lega\_drv\_rco\_cal

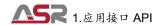
功能	校准 rco clock 接口	
函数定义	void lega_drv_rco_cal (void)	
参数	无	\ (7)\
返回	无	

### 1.1.31 lega\_wlan\_set\_country\_code

功能	配置国家码,Wi-Fi 协议栈会根据配置的国家码在规定信道工作	
函数定义	int lega_wlan_set_country_code(char *country)	
参数	● country 国家码字符串,当前支持: "CN" "EU" "JP" "US",如果没有配置国家码,默认 工作信道为 1-13 信道	
返回	<ul><li>0: 命令下达成功</li><li>-1: 参数 country 不合法</li><li>-2: 不支持该国家码</li></ul>	

### 1.1.32 lega\_wlan\_get\_country\_code

功能	获取当前配置的国家码
函数定义	char *lega_wlan_get_country_code(void)
参数	无
返回	国家码字符串: 当前支持: "CN" "EU" "JP" "US"



### 1.1.33 lega\_get\_client\_ip\_mac

功能	SOFTAP 模式下,获取连接到 softap 的对端的 MAC 地址与 ip 地址
函数定义	void lega_get_client_ip_mac(lega_wlan_ap_client_info_t* sta_addr)
参数	● sta_addr Wi-Fi 协议栈将相关信息填入 sta_addr 指向的内存空间,结构体定义详见 1.2 章节
返回	无

### 1.1.34 lega\_wlan\_err\_stat\_cb\_register

功能	注册 Wi-Fi 协议栈错误状态回调函数(用于 STATION 模式)
函数定义	int lega_wlan_err_stat_cb_register(lega_wlan_err_stat_handler fn)
参数	● lega_wlan_err_stat_handler fn Wi-Fi 协议栈检测到某些错误后调用此回调函数 回调函数类型定义如下: typedef void (*lega_wlan_err_stat_handler)(lega_wlan_err_status_e err_info) lega_wlan_err_status_e 结构体定义详见 1.2 章节
返回	0: 命令下达成功 非 0: 命令下达失败

### 1.1.35 lega\_wlan\_get\_wifi\_mode

功能	获取 Wi-Fi 工作模式
函数定义	int lega_wlan_get_wifi_mode(void)
参数	无
返回	1: STA MODE 2: SOFTAP MODE 3: SNIFFER MODE 0xff: Wi-Fi 未工作

### 1.1.36 lega\_wlan\_get\_softap\_info

功能	SOFTAP 模式下,获取 SOFTAP 配置的基本信息
函数定义	int lega_wlan_get_softap_info(lega_wlan_softap_info_t *ptr)
参数	● lega_wlan_softap_info_t *ptr Wi-Fi 协议栈将相关 SOFTAP 信息填入 ptr 指向的内存空间,包括已开启热点 的 ssid/ssid_len/chan/security 信息
返回	0: 命令下达成功 非 0: 命令下达失败



#### 1.1.37 lega\_wlan\_softap\_deauth\_peer

功能	SOFTAP 模式下,主动断开指定连接的客户端
函数定义	int lega_wlan_softap_deauth_peer(uint8_t *mac)
参数	● uint8_t *mac 要断开客户端的 mac address
返回	0: 命令下达成功 非 0: 命令下达失败



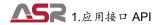


### 1.2 Wi-Fi 接口参数

上节 Wi-Fi 接口中涉及相关结构体参数在本节详述。

#### 1.2.1 Wi-Fi 开启参数

```
typedef struct {
    char
             wifi mode;
                                        /* refer to lega wifi type e */
    char
             security;
                                       /* security mode, refer to lega_wlan_security_e */
             wifi ssid[32];
                                       /* in station mode, indicate SSID of the wlan needs to be
    char
                                       connected.
                                        in softap mode, indicate softap SSID*/
                                       /* in station mode, indicate Security key of the wlan needs
    char
             wifi key[64];
                                        to be connected.
                                          in softap mode, indicate softap password. (Ignored in an
                                          open system.) */
             key_len;
    int
                                        /* Security key length */
             local_ip_addr[16];
                                       /* in softap mode to config ip for dut.
    char
                                        in station mode, to config static ip when set dhcp mode to
                                        WLAN DHCP DISABLE.*/
                                       /* in softap mode to config gateway for dut.
    char
             net_mask[16];
                                        in station mode, to config static ip net mask when set
                                        dhcp_mode to WLAN_DHCP_DISABLE.*/
    char
             gateway ip addr[16];
                                        /* in softap mode to config netmask for dut.
                                          in station mode, to config static ip gateway when set
                                            dhcp_mode to WLAN_DHCP_DISABLE.*/
                                       /* no use currently */
             dns_server_ip_addr[16];
    char
                                        /* start ip addr of dhcp pool in softap mode */
    char
             start_ip[16];
                                       /* end ip addr of dhcp pool in softap mode */
    char
             end ip[16];
             dhcp_mode;
                                       /* refer to lega_wlan_dhcp_mode_e */
    char
                                       /* softap channel in softap mode; connect channel in sta
    char
             channel;
                                        mode */
             mac_addr[6];
                                       /* connect bssid in sta mode */
    char
             reserved[32];
                                       /* no use currently */
    char
            wifi_retry_interval;
                                       /* no use currently */
    int
             wifi_retry_times;
                                       /* used in station mode to config reconnecting times after
    int
                                         disconnected */
                                      /* used in softap mode to config beacon listen interval */
    int
            interval:
                                       /* used in softap mode to config hidden SSID */
            hide;
    int
} lega wlan init type t;
typedef enum {
    STA=0x1, /* Act as a station which can connect to an access point */
    SOFTAP,
                /* Act as an access point, other station can connect, 4 stations Max */
                 /* Act as a sniffer */
    SNIFFER
} lega_wlan_type_e;
```



#### 1.2.2 Wi-Fi 获取 IP 信息参数

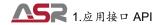
```
typedef struct {
                           /* start ip addr of dhcp pool in softap mode */
char
        start_ip[16];
                            /* end ip addr of dhcp pool in softap mode */
char
         end ip[16];
char
                            /* no use currently */
         dhcp;
                            /* mac address on the target wlan interface, "AA1122334455" */
char
      macaddr[16];
                           /* Local IP address on the target wlan interface, "192.168.1.100" */
char
        ip[16];
                            /* Router IP address on the target wlan interface, "192.168.1.1" */
char
        gate[16];
char
        mask[16];
                            /* Netmask on the target wlan interface, "255.255.255.0" */
char
        dns[16];
                            /* no use currently , ASCII */
                            /* no use currently , ASCII */
char
         broadcastip[16];
#ifdef LWIP_DUALSTACK
lega_ip6_addr_t ip6[3]
                            /* IPV6 IP address, number of per netif is 3 */
#endif
} lega_wlan_ip_stat_t;
```

#### 1.2.3 Wi-Fi 获取链路信息参数

```
typedef struct {
                              /* The link to wlan is established or not, 0: disconnected, 1:
    int is_connected;
connected. */
    int
            wifi_strength;
                             /* Signal strength of the current connected AP */
                             /* SSID ,max len:32, +1 is for '\0' when ssidlen is 32 */
    char
             ssid[32+1];
    char
                             /* BSSID of the current connected wlan */
             bssid[6]:
                             /* Channel of the current connected wlan */
    int
            channel;
} lega_wlan_link_stat_t;
```

#### 1.2.4 Wi-Fi 扫描完成回调函数参数

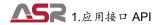
```
typedef struct {
                                 /* no use currently */
  uint8_t is_scan_adv;
  char ap_num;
                                 /* The number of access points found in scanning. */
  struct {
    char
             ssid[32+1];
                                 /* ssid max len:32. +1 is for '\0' when ssidlen is 32* /
    char
                                /* Signal strength, min:0, max:100. */
             ap_power;
    char
             bssid[6];
                                 /* The BSSID of an access point. */
                                 /* The RF frequency, 1-13 */
    char
             channel;
    uint8 t security;
                                 /* Security type, @ref wlan_sec_type_t */
  } * ap_list;
} lega_wlan_scan_result_t;
```



#### 1.2.5 Wi-Fi 关联完成获取 AP 信息回调函数参数

```
typedef enum {
    WLAN_SECURITY_OPEN,
                                                  //NONE
                                                  //WEP
    WLAN SECURITY WEP,
    WLAN_SECURITY_WPA,
                                                  //WPA
    WLAN SECURITY WPA2,
                                                  //WPA2
    WLAN_SECURITY_AUTO,
                                                  //WPA or WPA2
    WLAN_SECURITY_MAX,
}lega_wlan_security_e;
typedef struct {
    int
                               /* rssi */
            rssi;
                               /* ssid max len:32. +1 is for '\0' when ssidlen is 32
    char
            ssid[32+1];
                               /* pwd max len:64. +1 is for '\0' when pwdlen is 64 */
    char
            pwd[64+1];
                               /* BSSID of the wlan needs to be connected. */
    char
            bssid[6];
                               /* ssid length */
    char
            ssid len;
    char
            pwd_len;
                               /* password length */
                                /* wifi channel 0-13. */
            channel;
    char
                                / *refer to lega_wlan_security_e
    char
            security;
} lega_wlan_ap_info_adv_t;
```

#### 1.2.6 Wi-Fi 运行状态信息



#### 1.2.7 Wi-Fi 运行错误状态信息

```
/*WLAN error status*/
typedef enum {
    WLAN STA MODE BEACON LOSS = 1,
                                              //in sta mode, cannot receive beacon of peer
                                              connected AP for a long time
    WLAN STA_MODE_AUTH_FAIL,
                                              //in sta mode, connect fail during auth
    WLAN_STA_MODE_ASSOC_FAIL,
                                              //in sta mode, connect fail during association
    WLAN_STA_MODE_PASSWORD_ERR,
                                             //in sta mode, connect fail as password error
    WLAN_STA_MODE_NO_AP_FOUND,
                                              //in sta mode, connect fail as cannot find the
                                              connecting AP during scan
    WLAN STA MODE DHCP FAIL,
                                             //in sta mode, connect fail as dhcp fail
    WLAN_STA_MODE_CONN_RETRY_MAX,
                                              //in sta mode, connect fail as reach the max
                                              connect retry times
}lega_wlan_err_status_e;
```

#### 1.2.8 SOFTAP 模式下获取对端信息参数

```
* @brief sta ip and mac address used in softap mode
 * sta_ip_addr: e.g. when ip addr==192.168.1.1<-->sta_ip_addr == 0x0101A8C0
 */
typedef struct{
    uint32_t sta_ip_addr;
                                                           station ip addr */
    uint8 t sta mac addr[6];
                                                          /* station mac addr */
}lega_wlan_client_addr_info_t;
/*store linked station info*/
typedef struct{
    int client num;
                                                 /* linked station number */
    lega_wlan_client_addr_info_t sta[4];
                                                 /* linked station entry, max client number is 4 */
}lega_wlan_ap_client_info_t;
```



### 2.

# Wi-Fi 接口 API 用例

列举基本 Wi-Fi 操作的 API 调用流程。

### 2.1 Sta 模式的开启与关闭

- (1) 用户调用 lega\_wlan\_scan\_compeleted\_cb\_register 注册扫描完成回调函数。
- (2) 用户调用 lega\_wlan\_stat\_chg\_cb\_register 注册状态改变回调函数。
- (3) 用户调用 lega wlan ip got cb register 注册获取 IP 回调函数。
- (4) 用户调用 lega\_wlan\_open 开启 station 模式。
- (5) 扫描完成后, Wi-Fi 协议栈调用用户注册的扫描完成回调函数。
- (6) 连接成功后, Wi-Fi 协议栈调用用户注册的状态改变回调函数。
- (7) 获取 IP 地址后, Wi-Fi 协议栈调用用户注册的获取 IP 回调函数
- (8) 用户调用 lega\_wlan\_close 断开 AP 并关闭 station 模式。

### 2.2 Sniffer 模式的开启与关闭

- (1) 用户调用 lega\_wlan\_register\_monitor\_cb 注册 sniffer 模式接收回调函数。
- (2) 用户调用 lega\_wlan\_start\_monitor 开启 sniffer 模式。
- (3) 用户调用 lega\_wlan\_monitor\_set\_channel 配置 sniffer 模式的信道。
- (4) Wi-Fi 协议栈收到的包通过用户注册的回调函数报给用户。
- (5) 用户调用 lega\_wlan\_stop\_monitor 关闭 sniffer 模式。

### 2.3 Softap 模式的开启与关闭

- (1) 用户调用 lega wlan open 开启 softap 模式。
- (2) 其他 station 设备可以扫描到用户配置的 SSID 的 AP 并进行连接。
- (3) 用户调用 lega\_wlan\_close 断开所有连接的 station 设备并关闭 softap 模式。



3.

# 注意事项

- (1) Wi-Fi 协议栈现支持三种工作模式: Station 模式、Softap 模式、Sniffer 模式,但暂不支持 多模式共存,即在关闭一种模式前,无法开启另一种工作模式。
- (2) 对于 ASR582X 系列 FreeRTOS 通用 SDK,Wi-Fi 协议栈有三个优先级分别为 28、27、26 的 task,BLE 协议栈的优先级为 29,TCPIP 协议栈 task 默认设置的优先级为 27。为保证Wi-Fi 连接处理的实时性及优先性,一般情况下建议用户自定义的 task 的优先级低于Wi-Fi 相关 task。AT 命令 task 优先级为 20。
- (3) 用户注册的回调函数会跑在 Wi-Fi 线程中,为防止栈溢出等情况,建议回调函数实现尽可能简单。此外,在回调函数中不建议再直接调用 Wi-Fi 接口 API。

