



CV184X Wi-Fi 使用手册

Version: 1.0.0

Release date: 2025-03-10

©2025 北京晶视智能科技有限公司
本文件所含信息归北京晶视智能科技有限公司所有。
未经授权，严禁全部或部分复制或披露该等信息。

目录

1	声明	2
2	概述	3
3	配置说明	4
3.1	内核配置	4
3.2	设备树: wifisd 节点 (SDIO WiFi)	5
3.3	设备树: 勿删除 wifi-sd 节点	6
3.4	配置 SDIO	6
3.5	配置 Pinmux	6
3.6	配置 WIFI GPIO	7
3.7	menuconfig: 安装无线驱动 ko	7
4	Wi-Fi 工具	8
5	Wi-Fi 基本操作	11
5.1	STA 模式基本操作	11
5.1.1	加载驱动	11
5.1.2	启动 Wi-Fi 与连接 AP	12
5.1.3	关闭 Wi-Fi 与卸载驱动	14
5.2	SoftAP 模式基本操作	14
5.2.1	加载驱动	14
5.2.2	hostapd 配置, udhcpd 配置与启动 SoftAP	15
6	CV184x RTL8189FS WiFi 连接流程说明	16
6.1	编译与产物	16
6.2	板级 WiFi 连接流程	16
6.2.1	准备 wpa_supplicant、hostapd 与库 (开发机)	16
6.2.2	STA: wpa_supplicant 配置文件	17
6.2.3	STA: 板端操作步骤	17
6.2.4	STA: 验证	18
6.2.5	AP 模式: hostapd	18
6.2.6	AP 典型板端步骤 (经 eth0 做 NAT 上行)	19
7	测试	21
7.1	吞吐率测试	21
7.1.1	发送吞吐率测试	21
7.1.2	接收吞吐率测试	22

修订记录

Revision	Date	Description
1.0.0	2025/03/10	Initial version
1.0.1	2026/04/08	增补 RTL8189FS 连接流程与配置说明

1 声明



法律声明

本数据手册包含北京晶视智能科技有限公司（下称“晶视智能”）的保密信息。未经授权，禁止使用或披露本数据手册中包含的信息。如您未经授权披露全部或部分保密信息，导致晶视智能遭受任何损失或损害，您应对因之产生的损失/损害承担责任。

本文件内信息如有更改，恕不另行通知。晶视智能不对使用或依赖本文件所含信息承担任何责任。本数据手册和本文件所含的所有信息均按“原样”提供，无任何明示、暗示、法定或其他形式的保证。晶视智能特别声明未做任何适销性、非侵权性和特定用途适用性的默示保证，亦对本数据手册所使用、包含或提供的任何第三方的软件不提供任何保证；用户同意仅向该第三方寻求与此相关的任何保证索赔。此外，晶视智能亦不对任何其根据用户规格或符合特定标准或公开讨论而制作的可交付成果承担责任。

联系我们

地址 北京市海淀区丰豪东路 9 号院中关村集成电路设计园（ICPARK）1 号楼

深圳市宝安区福海街道展城社区会展湾云岸广场 T10 栋

电话 +86-10-57590723 +86-10-57590724

邮编 100094（北京）518100（深圳）

官方网站 <https://www.sophgo.com/>

技术论坛 <https://developer.sophgo.com/forum/index.html>

2 概述

Wi-Fi 是 Wi-Fi 联盟的商标, 为一种基于 IEEE 802.11 为标准的无线区域网路技术。具备 Wi-Fi 功能的移动终端可以在讯号覆盖范围内连上网际网路, 以减少架设电缆的困扰并提升使用的方便性。

目前已有很多处理器厂家提供各种型号的 Wi-Fi 处理器解决方案, 并有各自不同的驱动, 然这些驱动并不具备普适性。另外不同驱动版本所支持的功能及性能亦有可能会有差异, 需请 Wi-Fi 解决方案供应商提供合适的 Linux Wi-Fi 驱动来进行移植工作。

Linux 平台上对于不同 Wi-Fi 处理器的驱动与操作方式有通用性, 本文档会以 CV184X 为例分别介绍在不同接口上 (如 USB 或是 SDIO) 如何使用 Realtek 解决方案进行驱动移植与调适, 以及相关的操作。

针对 **RTL8189FS SDIO WiFi**** (如 mars37 工程中 cv1842 等板型) 的内核/defconfig、设备树要点、menuconfig 安装 ko、以及 ****STA (wpa_supplicant) / AP (hostapd)** 工具打包与板级连接流程, 详见[CV184x RTL8189FS WiFi 连接流程说明](#) 与 [配置说明](#) 中的相关小节。

本文所使用的 Wi-Fi 模组为

- AP6201BM (Broadcom bcm43013c1), 支持 SDIO 接口。
- Realtek **RTL8189FS**** (SDIO, 驱动模块名 ****8189fs.ko**), 板级流程见[CV184x RTL8189FS WiFi 连接流程说明](#)。

3 配置说明

本节说明内核 defconfig、设备树与 SDK menuconfig 中与 WiFi（含 RTL8189FS SDIO）相关的通用配置；板级工具打包与 STA/AP 操作命令见[CV184x RTL8189FS WiFi 连接流程说明](#)。

3.1 内核配置

在对应板级的 defconfig 中增加或确认下列选项，使内核支持 WiFi 引脚驱动、外置无线模块及 Realtek RTL8189FS（选项随芯片与方案略有差异，以实际 Kconfig 为准）。

示例路径（按板型替换）：

- build/boards/cv184x/cv1842hp_wevb_0014a_spinor/linux/
cvitek_cv1842hp_wevb_0014a_spinor_defconfig
- build/boards/cv184x/cv1842hp_wevb_0014a_emmc/linux/
cvitek_cv1842hp_wevb_0014a_emmc_defconfig

```
CONFIG_CVI_WIFI_PIN=y
CONFIG_CP_EXT_WIRELESS=y
CONFIG_WIRELESS=y
CONFIG_WLAN_VENDOR_REALTEK=y
CONFIG_RTL8189FS=m
CONFIG_WLAN=y
CONFIG_CFG80211=y
CONFIG_CFG80211_DEFAULT_PS=y
CONFIG_CFG80211_CRDA_SUPPORT=y
```

CONFIG_RTL8189FS=m 表示以模块方式编译，生成 8189fs.ko，由用户态 insmod 加载（亦可通过 SDK 选项自动安装到 rootfs，见下文 menuconfig）。

由于 Wi-Fi 接口为 SDIO，需在板级 DTS 与 asic dtsti 中正确使能 wifisd 与 wifi_pin 节点（见下文）。

3.2 设备树：wifisd 节点 (SDIO WiFi)

板级 DTS 位于各板型目录的 `dtb_arm` 下，通常已包含 `cv184x_base_arm.dtsi`、对应 `asic` 的 `cv184x_asic` 系列 `dtb`；WiFi SDIO 控制器节点定义在 default `dtb` 的 `cv184x_base.dtsi` 中（节点名 `wifisd`，`reg` 为 `wifi-sd@4320000`）。请确认节点 `status` 为 `okay`（若已 `okay` 则无需修改）。

SPI NOR 板型参考（字段以 SDK 实际为准）：

```
wifisd:wifi-sd@4320000 {
    compatible = "cvitek,cv184x-sdio";
    bus-width = <4>;
    reg = <0x0 0x4320000 0x0 0x1000>;
    reg_names = "core_mem";
    src-frequency = <375000000>;
    min-frequency = <400000>;
    max-frequency = <500000000>;
    64_addressing;
    reset_tx_rx_phy;
    non-removable;
    pll_index = <0x7>;
    pll_reg = <0x300207C>;
    no-mmc;
    no-sd;
    status = "okay";
};
```

eMMC 板型参考（`cv1842hp_wevb_0014a_emmc` 等）：

```
wifisd:wifi-sd@4320000 {
    compatible = "cvitek,cv184x-sdio";
    bus-width = <4>;
    reg = <0x0 0x4320000 0x0 0x1000>;
    reg_names = "core_mem";
    src-frequency = <400000000>;
    min-frequency = <400000>;
    max-frequency = <500000000>;
    64_addressing;
    reset_tx_rx_phy;
    non-removable;
    pll_index = <0x60>;
    pll_reg = <0x300209C>;
    clocks = <&clk CV184X_CLK_SD1>, <&clk CV184X_CLK_AXI4_SD1>;
    clock-names = "clk_wifisd", "clk_axi";
    no-mmc;
    no-sd;
    status = "okay";
};
```

3.3 设备树：勿删除 wifi-sd 节点

编辑 `build/boards/default/dts/cv184x/cv184x_asic_bga.dtsi` 或 `cv184x_asic_qfn.dtsi` 等当前板型使用的 `asic dtsi`。若存在 `/delete-node/ wifi-sd@4320000;`，请删除或注释该行，否则 WiFi SDIO 节点会被移除。示例如下：

```
/* /delete-node/ wifi-sd@4320000; */  
  
/delete-node/ i2c@04010000;  
/delete-node/ i2c@04020000;  
/delete-node/ ethernet@04520000;  
/delete-node/ i2s@04120000;  
...
```

3.4 配置 SDIO

请参考《CVITEK 外围设备驱动操作指南》中与 SDIO 相关章节。SDIO IO 电压为 3.3V，需确认 Wi-Fi 模块 IO 电压和 SDIO 电压一致。

3.5 配置 Pinmux

若 Wi-Fi 模块使用的接口为 SDIO 时，需配置 SDIO 的管脚复用。CV184X 可以通过在 `build/boards/{chip_name}/{board_name}/u-boot/cvi_board_init.c` 文件中添加以下 `pinmux` 设置来配置 SDIO 的 `pinmux`（这里是 `cv1843h` 的 `evb` 配置，具体配置哪根引脚需要查看电路图中 `soc` 到 `wifi` 模组 `processor_en` 的引脚是哪根）：

```
int cvi_board_init(void)  
{  
    ...  
    //#####WIFI  
    pinmux_config(PINMUX_SDIO1);  
    PINMUX_CONFIG(JTAG_CPU_TCK, XGPIOA_18);  
    ...  
    return 0;  
}
```

相关管脚配置细节，请参考 `u-boot-2021.10/board/cvitek/cv184x/board.c`

3.6 配置 WIFI GPIO

由于 Wi-Fi 模块的 `processor_en` 引脚是由 SOC 上的一根 GPIO 控制，为了操作这只 GPIO，我们专门做了一个简单的模块，在 wifi 驱动中使用该模块提供的接口对 wifi 进行上下电。可以通过设备树指定 wifi 使用的 gpio：（其中 `wakeup` 的功能没有使用，可以去除；`poweron` 引脚和上一小节设置的 `pinmux` 对应）

```
wifi_pin {
    compatible = "cvitek,wifi-pin";
    poweron-gpio = <&porta 18 GPIO_ACTIVE_HIGH>;
    wakeup-gpio = <&porte 7 GPIO_ACTIVE_HIGH>;
};
```

若板级在 `asic dtsi` 中以 `label` 引用节点，可采用如下形式（GPIO 以实际原理图为准）：

```
&wifi_pin {
    compatible = "cvitek,wifi-pin";
    poweron-gpio = <&porte 2 GPIO_ACTIVE_HIGH>;
    wakeup-gpio = <&porte 6 GPIO_ACTIVE_HIGH>;
};
```

该配置位于 `build/boards/default/dts` 下各芯片目录的 `base dtsi`（如 `cv184x_base.dtsi`）或板型所使用的 `asic dtsi` 中；`cv_i_wifi_pin` 驱动从 `poweron-gpio` 读取 WiFi 模组上电 GPIO，未配置则需在对应 `dts/dtsi` 中补齐。

3.7 menuconfig：安装无线驱动 ko

编译 SDK 时，可在 CViTek MediaSDK Configuration 的 SDK options 中勾选安装 `osdrv/extdrv/wireless` 目录下编译出的无线驱动 ko，使构建出的 `rootfs` 自动包含 `8189fs.ko` 等模块（路径以实际 `INSTALL_DIR` 为准）。

步骤简述：

1. 在 SDK 根目录执行 `make menuconfig`（或工程提供的配置命令）。
2. 进入 SDK options 子菜单（界面路径一般为 `Top → SDK options`，以实际菜单为准）。
3. 勾选“安装 `osdrv/extdrv/wireless` 下 ko”一类选项（英文菜单常含 `Install`、`wireless`、`ko` 等关键字，以界面实际文案为准），保存退出后重新编译并打包镜像。

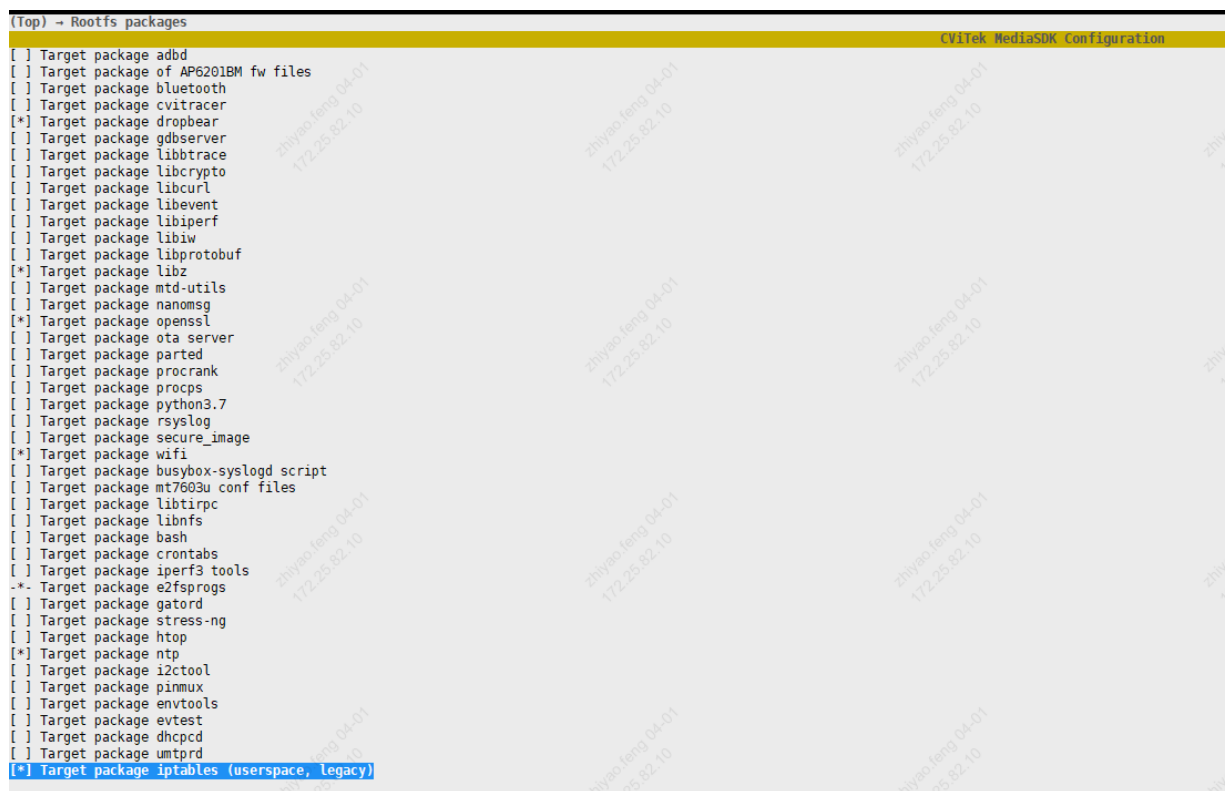
若仍采用手动拷贝 `8189fs.ko` 到板子，可不依赖该选项。

4 Wi-Fi 工具

操作 Wi-Fi 需要使用 wpa_supplicant、wpa_cli、hostapd 等开源工具。在 CViTek MediaSDK Configuration 中进入 Rootfs packages（菜单路径一般为 Top → Rootfs packages），按需勾选下列与无线及 AP/NAT 相关的包后保存配置，再编译镜像：

- Target package wifi：将 wpa_supplicant、hostapd 等打入 rootfs（进入该项后按需打开 wireless 子选项）。
- Target package iptables (userspace, legacy)：用户态 iptables；若需在板上做 NAT、转发（参见CV184x RTL8189FS WiFi 连接流程说明 中 AP 经 eth0 上行等步骤），请一并勾选。

下图为 Rootfs packages 中选中的部分示例（含 wifi、iptables 等；具体列表以你使用的 SDK 版本为准）：



- 亦可透过选单进入 Target package wifi 后打开 wireless 选项，存档离开后执行编译（与上文同属 Rootfs packages 菜单）。

```

Top) → Rootfs packages
CViTek MediaSDK Configuration
] Target adbd
] Target package of AP6201BM fw files
] Target package bluetooth
] Target package cvitracer
*] Target package dropbear
*] Target package gdbserver
] Target package libbtrace
*] Target package libcrypto
] Target package libcurl
] Target package libevent
] Target package libiperf
] Target package libiw
] Target package libprotobuf
*] Target package libz
*] Target package mtd-utils
] Target package nanomsg
] Target package openssl
*] Target package ota server
] Target package parted
] Target package procrank
] Target package procps
] Target package python3.7
] Target package rsyslog
] Target package secure_image
*] Target package wifi
*] Target package busybox-syslogd script
] Target package mt7603u conf files
] Target package libtirpc
] Target package libnfs
] Target package bash
]
]
[Space/Enter] Toggle/enter [ESC] Leave menu [S] Save
[O] Load [?] Symbol info [/] Jump to symbol
[F] Toggle show-help mode [C] Toggle show-name mode [A] Toggle show-all mode
[Q] Quit (prompts for save) [D] Save minimal config (advanced)

Top) → SDK options
CViTek MediaSDK Configuration
C library (glibc library for user mode application on arm) --->
[ ] Build static binary (no shared libs)
[*] Build SDK with debug config
[*] h264 vcode firmware
[*] h265 vcode firmware
[ ] Enable SDK sanitizer
[*] Do not install sample and self test application
[*] install the osdrv/extdrv/wireless/*.ko
[ ] Do not compile frame buffer drivers
[ ] Select CONFIG_NO_TP to build osdrv without Touchscreen driver(extdrv/tp)
[ ] Boot Time Optimization
[ ] Media Driver in Linux
[ ] Select CONFIG_USB_OSDRV_CVITEK_GADGET to build osdrv with usb gadget cvg
[*] Make the boot image only have one dtb
[*] enable the C906L DMA functions
[*] enable the audio drv to alios
[ ] enable 3a in alios
[ ]
[Space/Enter] Toggle/enter [ESC] Leave menu [S] Save
[O] Load [?] Symbol info [/] Jump to symbol
[F] Toggle show-help mode [C] Toggle show-name mode [A] Toggle show-all mode
[Q] Quit (prompts for save) [D] Save minimal config (advanced)

```

- 或修改 build/boards/{chip_name}/{board_name}/{board_name}_defconfig 使能如下选项，再透过命令模式，执行 defconfig \$CHIP_\$BOARD, 以自动做好配置

```

#
# Rootfs packages
#
...
CONFIG_TARGET_PACKAGE_WIFI=y

```

(下页继续)

(续上页)

```
CONFIG_CP_EXT_WIRELESS=y  
# end of Rootfs packages
```

若用户欲更新至最新版本, 可至 <http://w1.fi/releases> 或是 http://www.linuxfromscratch.org/blfs/view/svn/basicnet/wireless_tools.html 获取, 并自行安装至 rootfs。

SDK 预编译的 wpa_supplicant、hostapd 等二进制位于 ramdisk/rootfs/public/wifi/musl_arm/`` (与 ``packages_arch=musl_arm 一致) ; 从该目录打包拷贝到板子的步骤见CV184x RTL8189FS WiFi 连接流程说明。

5 Wi-Fi 基本操作

5.1 STA 模式基本操作

5.1.1 加载驱动

步骤 1. 载入驱动

需检查/mnt/system/ko/3rd 下是否已具备下面的文件，根据所用模组加载不同 ko:

```
[root@cvitek]/mnt/system/ko/3rd# ls
8188fu.ko  8189fs.ko
```

注意，由于 wifi 驱动并没有放在 linux 源码目录树中，所以无法 build-in，只能编译成 ko。

```
insmod /mnt/system/ko/3rd/8189fs.ko
或
insmod /mnt/system/ko/3rd/8188fu.ko
```

步骤 2. 查看驱动是否载入成功

执行 shell 指令:

```
ifconfig -a
```

若载入成功，可在执行 shell 指令后看到 wlan0 网口

```
/ # ifconfig -a
eth0      Link encap:Ethernet  HWaddr 00:00:00:00:00:00
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
          Interrupt:16

lo        Link encap:Local Loopback
          LOOPBACK  MTU:65536  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

sit0      Link encap:IPv6-in-IPv4
          NOARP  MTU:1480  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

wlan0     Link encap:Ethernet  HWaddr FC:6B:F0:7B:D1:29
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)
```

5.1.2 启动 Wi-Fi 与连接 AP

步骤 1. 启动 Wlan0 网口

sc 执行 shell 指令：

```
ifconfig wlan0 up
```

步骤 2. 启动 wpa_supplicant

执行 shell 指令：

```
echo "ctrl_interface=/var/run/wpa_supplicant" >/tmp/wpa_supplicant.conf
wpa_supplicant -iwlan0 -Dnl80211 -c/tmp/wpa_supplicant.conf &
```

- -iwlan0 表示使用 wlan0 接口
- -Dnl80211 表示使用 cfg80211 接口

步骤 3. 启动 wpa_cli

执行 shell 指令：

```
wpa_cli -i wlan0
```

执行成功会出现 “>” 提示符号

```
/ # wpa_cli
wpa_cli v2.6
Copyright (c) 2004-2016, Jouni Malinen <j@w1.fi> and contributors

This software may be distributed under the terms of the BSD license.
See README for more details.

Selected interface 'wlan0'

Interactive mode

>
```

步骤 4. 扫描附近 AP

在 “>” 提示符号后执行如下指令：

```
scan
```

在出现 “CTRL-EVENT-SCAN-RESULTS” 后, 再执行

```
scan_results
```

即可得到扫描结果

```
> scan
OK
[ 1206.695367] [0] RTW: wlan0- hw port(0) mac_addr =fc:6b:f0:7b:d1:29
[ 1206.794598] [0] RTW: nolinked power save leave
<3>CTRL-EVENT-SCAN-STARTED
<3>CTRL-EVENT-SCAN-RESULTS
<3>CTRL-EVENT-NETWORK-NOT-FOUND [1208.308629] [1] RTW: nolinked power save enter
FOUND
scan_results
> bssid / frequency / signal level / flags / ssid
ac:9e:17:5b:e7:8c      2462      -39      [WPA-PSK-COMP+TKIP][WPA2-PSK-COMP+TKIP][ESS]      SW-test
d8:fe:e3:9f:d8:d8      2427      -58      [WPA-PSK-COMP+TKIP][WPA2-PSK-COMP+TKIP][ESS]      avant
```

步骤 5. 连接 AP

- 连接配置为 WPA-PSK/WPA2-PSK 认证与加密类型方式的 AP

1. 在 “>” 提示符号后执行如下指令，以取得网路 ID（此示例中为 0）：

```
add_network
```

2. 配置网路的 SSID（此示例中的 SSID 为 SW-test，由步骤 4 取得）

```
set_network 0 ssid "SW-test"
```

3. 配置网路加密方式与密码（假设 SW-test 密码为 012345678）

```
set_network 0 psk "012345678"
```

4. 启动网路

```
select_network 0
```

5. 观察是否有收到 CTRL-EVENT-CONNECTED，若有，则表示连接成功。或可执行”status”指令，查询连线状态

```
> scan
OK
[ 1206.695367] [0] RTW: wlan0- hw port(0) mac_addr =fc:6b:f0:7b:d1:29
[ 1206.704508] [0] RTW: nolinked power save leave
<3>CTRL- EVENT-SCAN-STARTED
<3>CTRL- EVENT-SCAN-RESULTS
<3>CTRL- EVENT-NETWORK-NOT-[ 1208.308629] [1] RTW: nolinked power save enter
FOUND
scan_results
> bssid / frequency / signal level / flags / ssid
ac:9e:17:5b:e7:8c      2462      -39      [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS]      SW-test
d8:fe:e3:9f:d8:d8      2427      -58      [WPA-PSK-CCMP+TKIP][WPA2-PSK-CCMP+TKIP][ESS]      avant
```

6. 输入 “quit” 退出 wpa_cli, 执行 shell 指令如下以取得动态 IP 地址

```
udhcpc -b -i wlan0 -R &
```

7. 执行 ping 指令, 以观察网路是否正常运作, ex.

```
Ping 8.8.8.8
```

· 连接配置为 open system 的 AP

步骤与配置为 WPA-PSK/WPA2-PSK 认证与加密类型方式相同, 唯在配置网路加密方式需输入如下指令:

```
set_network 0 key_mgmt NONE
```

5.1.3 关闭 Wi-Fi 与卸载驱动

步骤 1. 执行 shell 指令如下:

```
ifconfig wlan0 down
```

步骤 2. 执行如下 shell 指令:

```
rmmod 8189fs.ko
或
rmmod 8188fu.ko
```

5.2 SoftAP 模式基本操作

5.2.1 加载驱动

载入驱动方式与 STA 模式相同, 请参考章节5.1.1. 加载驱动

5.2.2 hostapd 配置, udhcpd 配置与启动 SoftAP

启动 SoftAP 需先启动 hostapd。与 wpa_supplicant 类似, hostapd 可以用来配置 AP 端的各种认证协议, 以及连接流程。

步骤 1. 启动 hostapd。执行 shell 命令

```
ifconfig wlan0 192.168.1.1 up
hostapd /etc/network/hostapd.conf -B -i wlan0
```

步骤 2. 启动 udhcpd 以分配动态 IP 给连接装置, 执行 shell 命令

```
udhcpd /etc/network/udhcpd.conf
```

备注:

- 可透过修改 hostapd.conf 来配置 SoftAP 的 ssid, channel, 加密认证方式等, 文档位置在 SDK 包里的 /ramdisk/rootfs/overlay/{chip_name}/etc/network 或是平台上的 /etc/network。例如可修改 ssid 以及 wpa_passphrase 来配置 AP 名称以及登入密码

```
interface=wlan0
ctrl_interface=/var/run/hostapd
ssid=CV184X_EVB
channel=6
wpa=3
wpa_passphrase=012345678
```

其他参数意义可参考: <http://manpages.ubuntu.com/manpages/bionic/man5/udhcpd.conf.5.html>

- 可透过修改 udhcpd.conf 来配置 SoftAP 所分配的 IP 范围。udhcpd.conf 文档位置在 SDK 包里的 /ramdisk/rootfs/overlay/{chip_name}/etc/network 或是平台上的 /etc/network。

```
# The start and end of the IP lease block
start    192.168.1.10 #default: 192.168.0.20
end      192.168.1.254 #default: 192.168.0.254
```

6 CV184x RTL8189FS WiFi 连接流程说明

本文档说明在 cv1842 等 CV184x 平台启用 RTL8189FS SDIO WiFi 后的内核与设备树配置要点 (详见《配置说明》章节)、编译产物路径, 以及 STA (wpa_supplicant) 与 AP (hostapd) 用户态工具的打包与板级使用流程。

6.1 编译与产物

执行 build_osdrv (或工程等价构建命令) 后, 会生成 8189fs.ko。拷贝到 rootfs 的路径由 osdrv 的 INSTALL_DIR 决定, 例如:

```
install/soc_cv1842hp_wevb_0014a_emmc/rootfs/system/ko/3rd/8189fs.ko
```

将上述路径下的 8189fs.ko 拷贝到板子可访问目录 (如 /mnt/sd/)。若已在 menuconfig 中开启安装无线驱动 ko, 镜像内可能已包含该模块, 则无需再手动拷贝。

6.2 板级 WiFi 连接流程

6.2.1 准备 wpa_supplicant、hostapd 与库 (开发机)

来源说明: wpa_supplicant 与 hostapd 同源, 均来自 hostap 工程交叉编译后的安装树; SDK 中预置目录为 ramdisk/rootfs/public/wifi/musl_arm/ (与 packages_arch=musl_arm 一致)。无需单独再下载 hostapd 安装包, 与 wpa 相同, 从该目录拷贝二进制即可。

若 rootfs 内未带上述工具, 在开发机上打包时请注意: FAT 格式的 SD 卡不能建立符号链接, 需复制带版本号的 so, 再复制一份短文件名供动态链接器查找。

```
cd /path/to/your/sdk # 替换为实际 SDK 根目录
W=ramdisk/rootfs/public/wifi/musl_arm
rm -rf wifi_tools
mkdir -p wifi_tools
# STA
cp $W/usr/sbin/wpa_supplicant $W/usr/sbin/wpa_cli $W/usr/sbin/wpa_passphrase wifi_tools/
# AP (与 wpa 同目录, 同一次预编译产出)
```

(下页继续)

(续上页)

```
cp $W/usr/sbin/hostapd wifi_tools/
# 若存在可一并打包: hostapd_cli
test -f $W/usr/sbin/hostapd_cli && cp $W/usr/sbin/hostapd_cli wifi_tools/
cp $W/usr/lib/libnl-3.so.200.26.0 wifi_tools/
cp $W/usr/lib/libnl-genl-3.so.200.26.0 wifi_tools/
cp $W/usr/lib/libnl-route-3.so.200.26.0 wifi_tools/
cp wifi_tools/libnl-3.so.200.26.0 wifi_tools/libnl-3.so.200
cp wifi_tools/libnl-genl-3.so.200.26.0 wifi_tools/libnl-genl-3.so.200
cp wifi_tools/libnl-route-3.so.200.26.0 wifi_tools/libnl-route-3.so.200
cd wifi_tools
tar -cvf ../wifi_tools.tar .
```

将生成的 wifi_tools.tar 拷贝到板子可访问目录（如 U 盘/SD 的 /mnt/sd/）。

注解：文中 200.26.0 以你树内实际 usr/lib/libnl-*.so.* 文件名为准；若升级 libnl，需同步修改 cp 的源文件名。

6.2.2 STA: wpa_supplicant 配置文件

在板子可写目录创建 wpa_supplicant.conf（例如 /mnt/sd/wpa_supplicant.conf），内容示例（SSID/密码按实际修改）：

```
network={
    ssid="111"
    psk="1234qwer"
    key_mgmt=WPA-PSK
}
```

6.2.3 STA: 板端操作步骤

以下假设 wifi_tools.tar 与 wpa_supplicant.conf 均在 /mnt/sd/，且 8189fs.ko 已拷贝到板子（如 /mnt/sd/）。

```
# 1. 加载 WiFi 驱动
insmod /mnt/sd/8189fs.ko

# 2. 解压 wpa_supplicant、hostapd 与库
mkdir -p /mnt/sd/wifi
cd /mnt/sd/wifi
tar -xvf /mnt/sd/wifi_tools.tar

# 3. 设置库路径并启动 wpa_supplicant (STA)
export LD_LIBRARY_PATH=/mnt/sd/wifi
/mnt/sd/wifi/wpa_supplicant -B -i wlan0 -c /mnt/sd/wifi/wpa_supplicant.conf

# 4. 等待几秒后获取 IP
sleep 5
```

(下页继续)

(续上页)

```
udhcpc -i wlan0
```

5. 查看 wlan0 状态与 IP

```
ifconfig wlan0
```

若 udhcpc 长时间无响应，可检查热点是否开启 DHCP；部分热点网段为 192.168.43.x，可尝试手动配置：

```
ifconfig wlan0 192.168.43.100 netmask 255.255.255.0 up  
route add default gw 192.168.43.1
```

6.2.4 STA：验证

- ifconfig wlan0 应显示 inet addr 与 UP BROADCAST RUNNING。
- ping 192.168.43.1 （或热点网关）可验证链路；若热点共享网络，可再 ping 8.8.8.8 验证外网。

6.2.5 AP 模式：hostapd

- 二进制：与上一节相同，已从 \$W/usr/sbin/hostapd 打入 wifi_tools.tar，解压后位于 /mnt/sd/wifi/hostapd。
- 依赖库：与 wpa_supplicant 相同的 libnl 短名 so，设置 LD_LIBRARY_PATH=/mnt/sd/wifi 即可。

在板子 /mnt/sd/ 下创建 hostapd.conf （interface= 与实际 AP 网口一致；SSID/密码请按实际修改）：

```
interface=wlan1  
driver=nl80211  
ssid=MyAP  
hw_mode=g  
channel=6  
country_code=CN  
ieee80211d=1  
  
auth_algs=1  
wmm_enabled=1  
ignore_broadcast_ssid=0  
  
wpa=2  
wpa_passphrase=12345678  
wpa_key_mgmt=WPA-PSK  
wpa_pairwise=TKIP  
rsn_pairwise=CCMP  
ieee80211w=0
```

6.2.6 AP 典型板端步骤（经 eth0 做 NAT 上行）

典型场景：wlan0 曾作为 STA 连过热点（如 192.168.43.0/24），eth0 为有线或另一路上网（如 192.168.137.0/24）；在 wlan1 上开 AP，下挂终端经 wlan1 → NAT → eth0 出网。若 ip route 中存在经 wlan0 的 default 路由，会与“默认走 eth0 上行”冲突，可先删掉该默认路由（网关与接口以你板子 ip route 为准）。

udhcpd.conf（放在 /mnt/sd/udhcpd.conf；网段可自定义）：

```
start 192.168.1.100
end 192.168.1.200
interface wlan1
max_leases 20

option subnet 255.255.255.0
option router 192.168.1.1
option dns 8.8.8.8
```

执行以下命令配置转发规则：

```
export LD_LIBRARY_PATH=/mnt/sd/wifi

# 删除 STA 侧 default，仅保留经 eth0 的 default（上行口按实际修改）
ip route del default via 192.168.43.1 dev wlan0

# 开启 IPv4 转发
echo 1 > /proc/sys/net/ipv4/ip_forward

# NAT：若已存在则不再重复添加（上行口 eth0 按实际修改）
iptables -t nat -C POSTROUTING -o eth0 -j MASQUERADE 2>/dev/null || \
iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE

# 转发策略
iptables -P FORWARD ACCEPT
iptables -F FORWARD
iptables -A FORWARD -i wlan1 -j ACCEPT
iptables -A FORWARD -o wlan1 -j ACCEPT

# hostapd 需要运行时目录
mkdir -p /var/run/hostapd

killall wpa_supplicant 2>/dev/null

# AP 本机 IP 须与 udhcpd 中 option router 一致
ifconfig wlan1 192.168.1.1 netmask 255.255.255.0 up

/mnt/sd/wifi/hostapd -B /mnt/sd/hostapd.conf
udhcpd /mnt/sd/udhcpd.conf
```

udhcpd 说明：若系统无 udhcpd 需加入 rootfs 或改用 dnsmasq；前台调试可加 -f；部分环境需先创建 leases 文件（如执行 touch /var/lib/misc/udhcpd.leases，路径以本机 udhcpd -help 为准）。

无 DHCP 时：下挂设备可使用手工静态 IP（192.168.1.x，网关 192.168.1.1，掩码 255.255.255.0）。

若未做过 STA、本无 wlan0 的 default 路由，可省略 ip route del；192.168.43.1 与 wlan0 仅作示例。iptables 与 ip_forward 需内核打开 netfilter 且 rootfs 带 iptables。

停止：执行 `killall hostapd`；若跑了 `udhcpd` 则 `killall udhcpd`（或按 PID 结束）。

7 测试

7.1 吞吐率测试

Wifi 的性能可以透过吞吐率测试来观察与调校。一般吞吐率测试最常使用的工具为 iperf3。测试环境架设如下：



PC 通过有线的以太网路与无线路由器（wireless AP）连接，CVITEK 平台则透过 Wi-Fi 与无线路由器连接。假设在此示例中，PC 的 IP 地址为 192.168.0.11，CVITEK 平台的 IP 地址为 192.168.0.112。PC 与 CVITEK 平台皆有 iperf3 工具。

7.1.1 发送吞吐率测试

步骤 1. 在 PC 上进入 iperf3 工具目录，执行如下指令：

```
iperf3 -s
```

步骤 2. 平台执行 shell 指令如下：

- 测试 TCP 协定

```
iperf3 -c 192.168.0.11 -t 10
```

- 测试 UDP 协定

```
iperf3 -c 192.168.0.11 -t 10 -u -b 100M -l 32k
```

```
/ # iperf3 -c 192.168.0.11 -t 10
Connecting to host 192.168.0.11, port 5201
[ 5] local 192.168.0.112 port 50194 connected to 192.168.0.11 port 5201
[ ID] Interval      Transfer    Bitrate      Retr  Cwnd
[ 5]  0.00-1.00    sec   1.42 MBytes  11.9 Mbits/sec    0   138 KBytes
[ 5]  1.00-2.00    sec   941 KBytes   7.71 Mbits/sec   75   114 KBytes
[ 5]  2.00-3.00    sec   627 KBytes   5.14 Mbits/sec    0   130 KBytes
[ 5]  3.00-4.00    sec   941 KBytes   7.71 Mbits/sec    0   137 KBytes
[ 5]  4.00-5.00    sec   941 KBytes   7.71 Mbits/sec    0   138 KBytes
[ 5]  5.00-6.00    sec   1.23 MBytes  10.3 Mbits/sec    0   138 KBytes
[ 5]  6.00-7.00    sec   941 KBytes   7.71 Mbits/sec    0   140 KBytes
[ 5]  7.00-8.00    sec   1.53 MBytes  12.8 Mbits/sec    0   147 KBytes
[ 5]  8.00-9.00    sec   314 KBytes   2.57 Mbits/sec    1   158 KBytes
[ 5]  9.00-10.00   sec   753 KBytes   6.17 Mbits/sec    0   178 KBytes
-----
[ ID] Interval      Transfer    Bitrate      Retr
[ 5]  0.00-10.00   sec   9.51 MBytes  7.98 Mbits/sec   76
[ 5]  0.00-10.00   sec   8.89 MBytes  7.46 Mbits/sec

sender
receiver
```

透过 iperf3 发送测试可以取得结果如上图。各参数意义可以透过 iperf3 -h 取得说明。由上图可以观察到 10 秒平均吞吐率为 7.98Mbps。

7.1.2 接收吞吐率测试

步骤 1. 在平台上执行 shell 指令如下：

```
iperf3 -s
```

步骤 2. PC 上进入 iperf3 工具目录执行指令如下：

- 测试 TCP 协定

```
iperf3 -c 192.168.0.112 -t 10
```

- 测试 UDP 协定

```
iperf3 -c 192.168.0.112 -t 10 -u -b 100M -l 32k
```