WS73V100 Android 平台驱动移植

用户指南

文档版本　06

发布日期　2024-10-21

深圳市安信可科技有限公司

前言

概述

本文档介绍了 WS73V100 SDK 及在 Android 平台下的驱动移植指导，同时提供驱动适配的调试方法，帮助用户快速实现对驱动的移植。

读者对象

本文档主要适用于以下工程师：

- 软件开发工程师
- 软件测试工程师
- 技术支持工程师

符号约定

在本文中可能出现下列标志，它们所代表的含义如下。

符号	说明
⚠ 危险	表示如不避免则将会导致死亡或严重伤害的具有高等级风险的危害。
⚠ 警告	表示如不避免则可能导致死亡或严重伤害的具有中等级风险的危害。
⚠ 注意	表示如不避免则可能导致轻微或中度伤害的具有低等级风险的危害。

深圳市安信可科技有限公司

符号	说明
须知	用于传递设备或环境安全警示信息。如不避免则可能会导致设备损坏、数据丢失、设备性能降低或其它不可预知的结果。 “须知”不涉及人身伤害。
说明	对正文中重点信息的补充说明。 “说明”不是安全警示信息，不涉及人身、设备及环境伤害信息。

修改记录

文档版本	发布日期	修改说明
06	2024-10-21	• 更新“3.4.1 驱动移植步骤”章节内容。 • 更新“3.4.2 libbt-vendor.so 编译”章节内容。 • 更新“3.4.3 bluedroid 经典蓝牙屏蔽”章节内容。 • 更新“3.4.5 蓝牙业务调试”章节内容。
05	2024-06-27	更新“2.2 SDK 目录结构介绍”小节内容。
04	2024-02-26	• 更新“3.3.4 业务调试”小节内容。 • 更新“3.4.1 驱动移植步骤”小节内容。
03	2024-02-06	新增“3.4.6 星闪驱动编译加载”小节内容。
02	2024-01-15	更新“2.1 SDK 模块框架”小节内容。
01	2023-12-11	第一次正式版本发布。 • 更新“3.2 驱动加载步骤”小节内容。 • 更新“3.3.4 业务调试”小节内容。

深圳市安信可科技有限公司

文档版本	发布日期	修改说明
		• 更新“3.4.2 libbt-vendor.so 编译”小节内容。
00B02	2023-12-01	• 更新“2.2 SDK 目录结构介绍”小节内容。 • 更新“2.3.1 config 配置方法”、“2.3.2 驱动代码编译”小节内容。 • 新增“3.2 驱动加载步骤”小节内容。 • 更新“3.3.1 驱动编译配置”、“3.3.2 原生软件及库文件配置”小节内容。 • 更新“3.4.1 驱动移植步骤”、“3.4.2 libbt-vendor.so 编译”和“3.4.5 蓝牙业务调试”小节内容。
0B01	2023-11-07	第一次临时版本发布。

深圳市安信可科技有限公司

目 录

前言 i

1 开发环境搭建 1

1.1 SDK 开发环境简介 1

1.2 编译服务器环境要求 2

2 SDK 环境搭建 3

2.1 SDK 模块框架 3

2.2 SDK 目录结构介绍 4

2.3 编译 SDK 7

2.3.1 config 配置方法 7

2.3.2 驱动代码编译 10

3 Android 移植说明 11

3.1 单板配置文件 11

3.1.1 GPIO 配置 11

3.2 驱动加载步骤 11

3.3 Android 平台 WiFi 移植 12

3.3.1 驱动编译配置 12

3.3.2 原生软件及库文件配置 13

3.3.3 文件部署 15

3.3.4 业务调试 15

3.4 Android 平台蓝牙移植 18

3.4.1 驱动移植步骤 18

3.4.2 libbt-vendor.so 编译 19

3.4.3 bluedroid 经典蓝牙屏蔽 20

深圳市安信可科技有限公司

3.4.4 文件部署 20
3.4.4.1 Android7.0 及以下版本 20
3.4.4.2 Android8.0 及以上版本 20
3.4.5 蓝牙业务调试 20
3.4.6 星闪驱动编译加载 22
4 常见问题 24
4.1 USB 插入单板后未显示模组的 vid 和 pid 24
4.2 驱动加载出错 24
4.3 wpa_supplicant 启动失败 25

深圳市安信可科技有限公司

1 开发环境搭建

1.1 SDK 开发环境简介

1.2 编译服务器环境要求

1.1 SDK 开发环境简介

典型的 SDK 开发环境主要包括：

- Linux 服务器

  Linux 服务器主要用于建立交叉编译环境，实现在 Linux 服务器上编译出可以在目标板上运行的可执行代码。

- PC 工作台

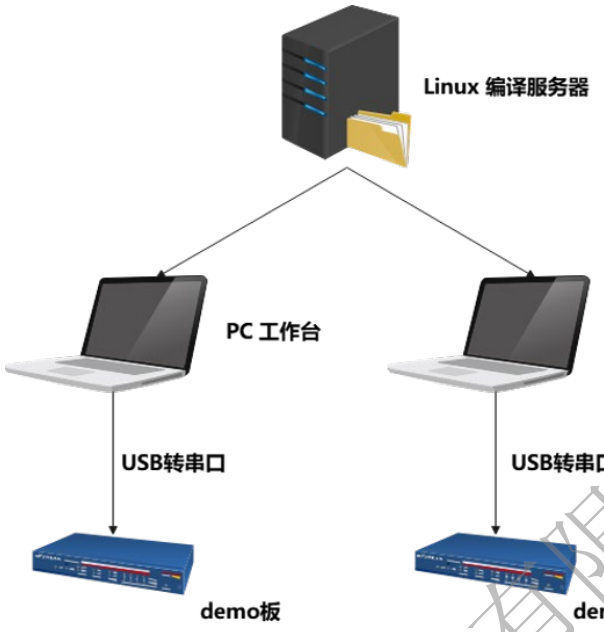
  工作台主要用于目标板烧录和调试，通过串口与目标板连接，开发人员可以在工作台中烧录目标板的镜像、调试程序。工作台通常需要安装终端工具，用于登录 Linux 服务器和目标板，查看目标板的打印输出信息。工作台一般为 Windows 或 Linux 操作系统，在 Windows 或 Linux 工作台运行的终端工具通常有 SecureCRT、Putty、miniCom 等，这些软件需要从其官网下载。

- 目标板

  本文的目标板以 demo 板为例，demo 板与工作台通过 USB 转串口连接。工作台将交叉编译出来的 demo 板镜像通过串口烧录到 demo 板，如图 1-1 所示。

深圳市安信可科技有限公司

图1-1 SDK 开发环境



1.2 编译服务器环境要求

表1-1 系统软件

系统软件	版本要求
python3	3.8+
realpath	8.0+

表1-2 Python 依赖库

依赖库	版本要求	安装命令
pycparser	2.21+	pip3 install pycparser>=2.21

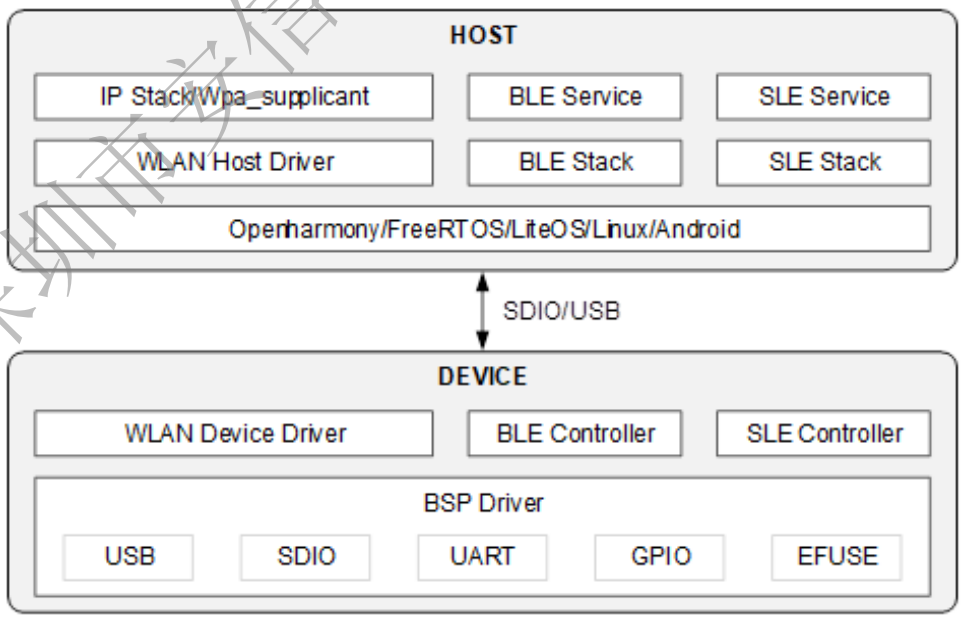
深圳市安信可科技有限公司

2 SDK 环境搭建

2.1 SDK 模块框架
2.2 SDK 目录结构介绍
2.3 编译 SDK

2.1 SDK 模块框架

图2-1 WS73 SDK 系统框架



深圳市安信可科技有限公司

表2-1 WS73 SDK 系统框架说明

序号	模块名	功能介绍
1	IP Stack	提供网络通信服务，由主控侧提供，WS73 提供功能适配。
2	Wpa_supplicant	提供 Wi-Fi 通信服务，由开源代码仓提供，WS73 提供功能适配。
3	BLE Service	提供 BLE 通信服务，由开源代码仓提供，WS73 提供功能适配。
4	SLE Service	提供星闪通信服务，由 WS73 提供。
5	WLAN Host Driver	支持 Wi-Fi 服务的驱动，由 WS73 提供。
6	BLE Stack	支持 BLE 服务的驱动，由 WS73 提供。
7	SLE Stack	支持 SLE 服务的驱动，由 WS73 提供。
8	WLAN Device Driver	支持 Wi-Fi 服务的固件，由 WS73 提供。
9	BLE Controller	支持 BLE 服务的固件，由 WS73 提供。
10	SLE Controller	支持 SLE 服务的固件，由 WS73 提供。

2.2 SDK 目录结构介绍

说明

SDK 中 firmware 具体差异可参考《NW117V100 系列 Wi-Fi、BLE 和 SLE Combo 芯片 用户指南》 产品概述小节。

SDK 目录结构如下：

```
.
├── application              # 用户态二进制文件和示例代码
│   ├── bin                        # 星闪可执行程序
│   ├── lib                          # 星闪可执行程序依赖库
│   ├── sample               # WS73 BLE/SLE示例代码
├── build                    # SDK构建所需配置文件和编译脚本
```

深圳市安信可科技有限公司

```
|   ├── config                          # WS73 默认配置文件
├── driver                      # WS73的驱动文件
|   ├── android_autoconfig.h      # Android配置文件
|   ├── android.config        # Android配置文件
|   ├── bsle              # WS73 BLE/SLE驱动文件
|   ├── Makefile              # WS73 驱动编译入口
|   ├── platform              # WS73 平台文件
|   └── wifi              # WS73 WiFi驱动文件
├── firmware                    # WS73 驱动依赖固件
|   ├── e                   # USB2.0/UART版本固件
|   └── us                  # SDIO/USB版本固件
├── include                     # API头文件存放目录

├── Makefile                    # Makefile编译总入口
├── open_source                 # 存放开源组件及相关patch
├── output                      # 编译时生成的目标文件和中间文件
```

SDK 开源目录代码说明：

表2-2 WS73 开源代码目录说明

序号	模块名	文件名	功能介绍
1	ble_gatt_client	ble_gatt_client.c	ble gatt 协议客户端 sample 代码，主要实现客户端发现所有服务端注册服务。
2	ble_uuid_server	ble_server_adv.c	ble gatt 协议服务端 sample 代码，主要实现服务端设置广播数据和参数，启动广播等功能。
3	ble_uuid_server	ble_uuid_server.c	ble gatt 协议服务端 sample 代码，主要实现注册 uuid 服务和数传等功能。
4	ble_hid_keyboa	ble_hid_keyboard	ble 键盘 hid 服务 sample 代码，主

深圳市安信可科技有限公司

序号	模块名	文件名	功能介绍
	rd_server	_server.c	要实现键盘 hid 服务注册，广播和数传等功能。
5	ble_hid_mouse_server	ble_hid_mouse_server.c	ble 鼠标 hid 服务 sample 代码，主要实现实现鼠标 hid 服务注册，广播和数传等功能。
6	sle_mouse_server	sle_mouse_adv.c	星闪 sle 鼠标广播 sample 代码，主要实现鼠标广播参数和数据设置，启动广播等功能。
7	sle_mouse_server	sle_mouse_dis_server.c	星闪 sle 鼠标服务注册 sample 代码，主要实现鼠标服务注册等功能。
8	sle_mouse_server	sle_mouse_hid_server.c	星闪 sle 鼠标 hid 服务注册 sample 代码，主要实现鼠标 hid 服务注册等功能。
9	sle_uuid_client	sle_uuid_client.c	星闪 ssap 协议客户端 sample 代码，主要实现 ssap 协议客户端初始化和发现设备等功能。
10	sle_uuid_server	sle_server_adv.c	星闪 ssap 协议服务端广播 sample 代码，主要实现服务端广播参数设置和广播启动等功能。
11	sle_uuid_server	sle_uuid_server.c	星闪 ssap 协议服务端 sample 代码，主要实现服务端服务注册和数传等功能。

深圳市安信可科技有限公司

2.3 编译 SDK

2.3.1 config 配置方法

首次运行

用户首次运行时，SDK 根目录下执行“vi build/config/ws73_default.config”命令，配置相关参数。

编译参数

WS73 编译相关参数如表 2-3 所示。

表2-3 编译参数配置说明

参数	说明	示例
WSCFG_USING_GCC	是否配置编译器类型为 GCC	WSCFG_USING_GCC=y
WSCFG_CROSS_COMPILE	配置编译器存放路径	WSCFG_CROSS_COMPILE=/opt/toolchains/bin/arm-himix100-linux-
WSCFG_KERNEL_DIR	配置内核存放路径	WSCFG_KERNEL_DIR=/usr/kernel/linux-4.9.y
WSCFG_ARCH_ARM	是否配置 CPU 架构类型为 ARM	WSCFG_ARCH_ARM=y
WSCFG_BUS_SDIO	是否配置通信总线类型为 SDIO	WSCFG_BUS_SDIO=y

深圳市安信达科技有限公司

图2-2 编译参数配置示例

```
#
# Configure the compilation environment
#
WSCFG_USING_GCC=y
# WSCFG_USING_LLVM_CLANG is not set
WSCFG_CROSS_COMPILE="/opt/toolchains/[redacted]/bin/arm-himix100-linux-"
WSCFG_CLANG_EXTRA_CFLAGS=""
WSCFG_KERNEL_DIR="/home/share/[redacted]/kernel/linux-4.9.y"
WSCFG_ARCH_ARM=y
# WSCFG_ARCH_CUSTOM is not set
WSCFG_ARCH_NAME="arm"
BOARD_ASIC=y
BOARD_ASIC_WIFI=y
# WSCFG_BUS_SDIO
WSCFG_BUS_USB=y
# WSCFG_BUS_UART is not set
WSCFG_LINUX=y
_PRE_OS_VERSION_LINUX=0
_PRE_OS_VERSION_WIN32=1
_PRE_OS_VERSION_WINDOWS=2
_PRE_OS_VERSION_RAW=3
_PRE_OS_VERSION_HiRTOS=4
_PRE_OS_VERSION_WIN32_RAW=5
_PRE_OS_VERSION_LITEOS=6
_PRE_OS_VERSION_ANDROID=7
_PRE_OS_VERSION=0
# end of build
```

说明

- WSCFG_CROSS_COMPILE 和 WSCFG_KERNEL_DIR 需读者依照实际情况进行配置。
- 若非 ARM 架构，可关闭 WSCFG_ARCH_ARM，打开 WSCFG_ARCH_CUSTOM 宏，并修改 WSCFG_ARCH_NAME，如 WSCFG_ARCH_NAME="mips"。
- 若非 SDIO 总线，可关闭 WSCFG_BUS_SDIO，打开对应总线宏，如 WSCFG_BUS_USB。

平台参数

WS73 平台相关参数如表 2-4 所示。

表2-4 平台参数配置说明

参数	说明	示例
CONFIG_FIRMWARE_BIN_PATH	配置 ws73.bin 文件路径	CONFIG_FIRMWARE_BIN_PATH="/etc/ws73/ws73.bin"
CONFIG_FIRMWARE_WIFICALI_PATH	配置 wifi_cali.bin 文件路径	CONFIG_FIRMWARE_WIFICALI_PATH="/etc/ws73/wifi_cali.bin"
CONFIG_FIRMWARE_BSLECALI_PATH	配置 btc_cali.bin 文件路径	CONFIG_FIRMWARE_BSLECALI_PATH="/etc/ws73/btc_cali.bin"
CONFIG_FIRMWARE_WOW_PATH	配置 wow.bin 文件路径	CONFIG_FIRMWARE_WOW_PATH="/etc/ws73/wow.bin"
CONFIG_INI_FILE_PATH	配置 ws73_cfg.ini 文件路径	CONFIG_INI_FILE_PATH="/etc/ws73_cfg.ini"

深圳市安信可科技有限公司

参数	说明	示例
CONFIG_PLAT_DFR_OUTPUT_PATH	配置 ws73 panic 文件存在路径	CONFIG_PLAT_DFR_OUTPUT_PATH="/etc/ws73"

图2-3 平台参数配置示例

```
#
# Platform
#
_PRE_PLAT_SHA256SUM_CHECK=y
WSCFG_ONEIMAGE=y
WSCFG_PLAT_VARIABLE_FEATRUE=y
WSCFG_PLAT_DIAG_LOG_OUT=y
# CONFIG_DIAG_SUPPORT_SOCKET is not set
CONFIG_DIAG_SUPPORT_UART=y
CONFIG_DIAG_SUPPORT_LOCAL_LOG=y
CONFIG_PLAT_SUPPORT_DFR=y
CONFIG_PLAT_DFR_OUTPUT_PATH="/etc/ws73"
CONFIG_HCC_SUPPORT_TEST=y
CONFIG_HCC_ERROR_PRINT=y
_PRE_PLAT_HCC_SDIO=y
BT_EM_BUFFER_CALI_SUPPORT=y
# _PRE_PLAT_SLP_UART_FORWARD is not set
CONFIG_DFX_SUPPORT_SYSFS=y
# CONFIG_SDIO_RESCAN is not set

#
# Configure the loading path of files such as firmware
#
CONFIG_FIRMWARE_BIN_PATH="/etc/ws73/ws73.bin"
CONFIG_FIRMWARE_WIFICALI_PATH="/etc/ws73/wifi_cali.bin"
CONFIG_FIRMWARE_BSLECALI_PATH="/etc/ws73/btc_cali.bin"
CONFIG_FIRMWARE_WOW_PATH="/etc/ws73/wow.bin"
CONFIG_INI_FILE_PATH="/etc/ws73_cfg.ini"
# end of Configure the loading path of files such as firmware
```

说明

- 配置 CONFIG_FIRMWARE_BIN_PATH、CONFIG_FIRMWARE_WIFICALI_PATH、CONFIG_FIRMWARE_BSLECALI_PATH、CONFIG_FIRMWARE_WOW_PATH等参数后，在加载 ko 前，需要保证单板对应目录下，有对应的二进制文件。二进制文件在 SDK 中的 firmware 目录下，有 us/e 两个版本，根据实际情况选用对应版本。两个版本的具体差异可参考《NW117V100 系列 Wi-Fi、BLE 和 SLE Combo 芯片 用户指南》 产品概述小节。
- 配置 CONFIG_INI_FILE_PATH 后，在加载 ko 前，需要保证单板对应目录下，有对应的 ini 文件。
- 配置 CONFIG_PLAT_DFR_OUTPUT_PATH 需要保证该路径随时可写，以便出现系统 panic 时，能正常保存日志

重复运行

用户执行“make android”完成编译后，可以在根目录下执行“vi build/config/ws73_default.config”，重新配置相关参数。

深圳市安凯科技有限公司

2.3.2 驱动代码编译

根目录下执行“make android”指令运行脚本编译，即可编译出对应的 SDK 程序。编译命令列如表 2-5 所示。

表2-5 make 参数列表

参数	示例	说明
无	make	执行 Linux 系统全量编译。
android	make android	执行 Android 系统全量编译。
clean	make clean	清理编译过程中生成的中间文件和烧写文件。
hso	make hso	生成 DebugKit 应用数据库，用于驱动维测功能

深圳市安信可科技有限公司

3 Android 移植说明

3.1 单板配置文件
3.2 驱动加载步骤
3.3 Android 平台 WiFi 移植
3.4 Android 平台蓝牙移植

3.1 单板配置文件

3.1.1 GPIO 配置

具体内容请参见《WS73V100 单板配置文件说明书》中“GPIO 配置”小节。

3.2 驱动加载步骤

步骤 1 默认加载 cfg80211.ko

步骤 2 确认采用的驱动加载模式，若通过配置启动脚本来加载驱动，可直接跳到步骤 4；若通过适配驱动模组设备号，再加载驱动，可继续执行步骤 3。

步骤 3 针对 USB 模组，可执行“lsusb”查看模组的 VID 和 PID 号，其中 WS73 模组默认值为 ffff:3733。

```
/komod # lsusb
Bus 001 Device 001: ID 1d6b:0002
Bus 001 Device 002: ID ffff:3733
Bus 002 Device 001: ID 1d6b:0003
```

深圳市安信可科技有限公司

针对 SDIO 模组，可查看内核实际注册的设备节点来获取模组的 VID 和 PID 号，其中 WS73 模组默认值为 ffff:3733。

```
/ # cat /sys/bus/sdio/devices/mmc1\:0001\:1/uevent
SDIO_CLASS=07
SDIO_ID=FFFF:3733
MODALIAS=sdio:c07vFFFFd3733
/ #
```

步骤 4 先加载 plat_soc.ko，再根据需要加载 wifi_soc.ko、sle_soc.ko、ble_soc.ko（wifi、sle、ble 间无依赖关系，只要在 plat ko 后加载即可）。

----结束

3.3 Android 平台 WiFi 移植

3.3.1 驱动编译配置

说明

- 编译工具链接需要提前配置环境变量，例如 export PATH=$PATH:/home/wifi/share/arm-linux-gnueabihf/bin，路径根据实际情况配置。
- 使用不同工具链时，可能会导致编译过程中出现告警导致编译失败，可在相关驱动组件的 Makefile 文件中修改编译选项。

步骤 1 解压驱动源码包并配置编译参数。

将驱动源码包 WS73V100.tar.gz 置于服务器并进行解压，参考 2.3 编译 SDK 配置编译工具链、内核路径、CPU 架构等，命令如下：

```
$ tar -xzf WS73V100.tar.gz
$ cd WS73V100
```

步骤 2 完成配置后，执行 make all 命令编译驱动文件，在 output 生成目标文件：

```
$ make android
```

编译结果输出到"output"目录下，如表 3-1 所示。

表3-1 全量编译结果

文件名	说明
plat_soc.ko	WS73 平台驱动模块。

深圳市安信可科技有限公司

文件名	说明
wifi_soc.ko	WS73WiFi 驱动模块。
ble_soc.ko	WS73 蓝牙驱动模块。
sle_soc.ko	WS73 星闪驱动模块。
ws73_cfg.ini	WS73 客制化的配置文件。

----结束

3.3.2 原生软件及库文件配置

说明

- 通过 3.3.1 驱动编译配置已完成驱动的编译，可参考 3.3.4 业务调试直接加载驱动，验证驱动功能是否正常。
- 本章以 Android 10 平台为例，介绍将 WS73 驱动代码移植到 Android 编译框架的方法，其他 Android 平台移植步骤与 Android 10 平台类似。各厂家文件路径存在差异，具体移植流程需要咨询对应主控平台厂家。
- {vendor}：主控平台厂家名
- {product}：主控平台厂家产品名

步骤 1 进入 device/{vendor}/{product}/BoardConfig.mk，添加如下原生修改：

```
WPA_SUPPLICANT_VERSION := VER_0_8_X
BOARD_WPA_SUPPLICANT_DRIVER := NL80211
BOARD_WPA_SUPPLICANT_PRIVATE_LIB := lib_driver_cmd_bcmdhd
BOARD_HOSTAPD_DRIVER := NL80211
BOARD_HOSTAPD_PRIVATE_LIB := lib_driver_cmd_bcmdhd
BOARD_WLAN_DEVICE := bcmdhd
```

说明

- 目前 Android 系统中 wpa_supplicant 软件对应的源码路径：external/wpa_supplicant_8，该软件使用的原生库文件为 libwifi-hal.so，对应 Android 源码路径：frameworks/opt/net/wifi/libwifi_hal。
- 一般用户都会按照以上标识，写明使用的具体 wpa_supplicant 配置。如果不清楚用户是否真的使用原生配置，可在用户的代码路径 device/{vendor}/{product}下，搜索“BOARD_WPA_SUPPLICANT_DRIVER ”，打开对应的文件，查看用户使用的具体配置及 lib 库，可执行“grep -rn BOARD_WPA_SUPPLICANT_DRIVER”进行搜索。

步骤 2 进入 Android 根目录，可选择整编 Android 镜像或进入对应目录进行各模块编译：

```
# 整体编译
$ source build/envsetup.sh
$ lunch {product}
$ make -j32

# 模块编译
$ source build/envsetup.sh
$ lunch {product}

# 编译wpa_supplicant对应的静态库lib_driver_cmd_bcmdhd.a
$ cd hardware/broadcom/wlan/bcmdhd/wpa_supplicant_8_lib
$ mm -j8

# 编译wpa_supplicant可执行文件
$ cd -
$ cd external/wpa_supplicant_8
$ mm -j8

# 编译libwifi-hal对应的静态库libwifi-hal-bcm.a
$ cd hardware/broadcom/wlan/bcmdhd/wifi_hal
$ mm -j8

# 编译libwifi-hal.so
$ cd -
$ cd frameworks/opt/net/wifi/libwifi_hal
$ mm -j8
```

步骤 3 获取原生软件及库文件：

- wpa_supplicant 文件一般生成在 out/target/product/{product}/system/bin/或 out/target/product/{product}/vendor/bin/hw/目录下。对应单板上的路径为 system/bin 或/vendor/bin/hw。
- libwifi-hal.so 文件一般生成在 out/target/product/{product}/vendor/lib64/或 out/target/product/{product}/vendor/lib/目录下。对应单板上的路径为 vendor/lib64 或 vendor/lib。

----结束

深圳市安信可科技有限公司

3.3.3 文件部署

根据平台上所使用的路径进行文件存放即可。一般情况为：

- Android7.0 及以下版本使用的是/system 路径。
- Android8.0 及以上版本使用的是/vendor 路径。

Android7.0 及以下版本

- 驱动 ko 的路径会存放在/system/lib/modules 或/system/lib64/modules 目录下。
- wpa_supplicant 会存放在/system/bin 目录下。
- firmware 文件会存放在/system/etc/firmware 目录下。此路径与平台 uboot 中 firmware_class.path 的设定有关，实际需要根据主控平台的路径确定。
- ini 配置文件会存放在/system/etc/wifi 目录下。

Android8.0 及以上版本

- 驱动 ko 的路径会存放在/vendor/lib/modules 或/vendor/lib64/modules 目录下。
- wpa_supplicant 会存放在/vendor/bin/hw 目录下。
- firmware 文件会存放在/vendor/firmware 目录下。此路径与平台 uboot 中 firmware_class.path 的设定有关，实际需要根据主控平台的路径确定。
- ini 配置文件会存放在/vendor/etc/wifi 目录下。

3.3.4 业务调试

步骤 1 将 WS73 所需的文件放至单板对应目录下，其中固件文件在 firmware 目录下，驱动文件在 output 目录下，如表 3-2 所示。

表3-2 驱动加载所需文件及存放路径

文件	存放路径（默认）
ws73.bin	/etc/ws73
wifi_cali.bin	/etc/ws73
btc_cali.bin	/etc/ws73
wow.bin	/etc/ws73
ws73_cfg.ini	/etc
plat_soc.ko	/vendor/lib/modules

深圳市安信可科技有限公司

文件	存放路径（默认）
wifi_soc.ko	/vendor/lib/modules

步骤 2 加载 Wi-Fi 驱动文件，命令如下：

```
insmod /vendor/lib/modules/plat_soc.ko
insmod /vendor/lib/modules/wifi_soc.ko
```

步骤 3 启动 wpa_supplicant，命令如下：

```
/vendor/bin/hw/wpa_supplicant -ip2p0 -Dnl80211 -c/data/vendor/wifi/wpa/p2p_supplicant.conf -puse_p2p_group_interface=0 -N -iwlan0 -Dnl80211 -c/data/vendor/wifi/wpa/wpa_supplicant.conf -O/data/vendor/wifi/wpa/sockets &
```

说明

- -O/data/vendor/wifi/wpa/sockets 也需要根据主控平台进行配置，一般在 /data/vendor/wifi/wpa/sockets 或/data/misc/wifi/sockets 目录下。(注意：O，O 为大写)
- wpa_supplicant.conf 与 p2p_supplicant.conf 的文件路径需要根据主控平台路径进行配置，一般在/data/vendor/wifi/wpa 或/data/misc/wifi/目录下。

步骤 4 启动 wpa_cli，进行 Wi-Fi 扫描链接，命令如下：

- 启动 wpa_cli

  ```
  wpa_cli -iwlan0 -p/etc/Wireless/wpa_supplicant
  ```

- 启动 wpa_cli，执行扫描。

  a. 在“>”后执行“scan”命令，驱动扫描流程。

  b. 收到“CTRL-EVENT-SCAN-RESULTS”后，执行“scan_results”，获得扫描结果。

  ```
  > scan
  > scan_results
  ```

- 启动 wpa_cli，执行连接。

  a. 在“>”后执行“add_network”命令，该命令会返回一个数字，表示添加的网络 ID 号。

  b. 执行“set_network 网络 id ssid “AP 的 SSID”” 命令，配置网络 ID 的 SSID，其中网络 id 默认为 0。

  c. 执行“set_network 网络 id key_mgmt NONE”命令，配置网络 ID 的加密方式。

  d. 执行“select_network 网络 id”命令，选择并网络 ID 进行连接。

深圳市安信可科技有限公司

e. 收到“CTRL-EVENT-CONNECTED”表示连接成功。

```
> add_network
> set_network 0 ssid “sta-test”
> set_network 0 key_mgmt WPA-PSK
> set_network 0 psk “12345678”
> select_network 0
> q
```

图3-1 STA 链接过程



步骤 5 配置 Android 原生流程

若步骤 1~步骤 4 均执行成功，则证明 Wi-Fi 驱动功能正常，接下来可以对 Android Wi-Fi 启动流程进行配置。

1. 根据主控平台，在 Android 上进行编译时，将步骤 1 中的驱动文件拷贝到步骤 1 中单板对应的路径下。
2. 根据主控平台，在 Android 上启动流程中加载 ko；一般可把步骤 2 加载驱动操作写到 device/**{vendor}**/**{product}**/init.**{product}**.rc 文件中。
3. 根据主控平台，在 Android 上启动 wpa_supplicant。一般可把步骤 3 修改成如下规则，再写到 device/**{vendor}**/**{product}**/init.**{product}**.rc 文件中。

```
service wpa_supplicant /vendor/bin/hw/wpa_supplicant \
    -ip2p0 -Dnl80211 -c/data/vendor/wifi/wpa/p2p_supplicant.conf \
    -e/data/misc/wifi/entropy.bin -puse_p2p_group_interface=0 -N \
    -iwlan0 -Dnl80211 -c/data/vendor/wifi/wpa/wpa_supplicant.conf \
    -O/data/vendor/wifi/wpa/sockets \
```

```
-g@android:wpa_wlan0
interface android.hardware.wifi.supplicant@1.0::ISupplicant default
interface android.hardware.wifi.supplicant@1.1::ISupplicant default
interface android.hardware.wifi.supplicant@1.2::ISupplicant default
class main
socket wpa_wlan0 dgram 660 wifi wifi
disabled
oneshot
```

步骤 6 根据上述修改，编译出 Android 镜像，通过设置界面连接 Wi-Fi 即可。

----结束

3.4 Android 平台蓝牙移植

3.4.1 驱动移植步骤

说明

- 编译工具链接需要提前配置环境变量，例如 export PATH=$PATH:/home/wifi/share/arm-linux-gnueabihf/bin，路径根据实际情况配置。
- 使用不同工具链时，可能会导致编译过程中出现告警导致编译失败，可在相关驱动组件的 Makefile 文件中修改编译选项。

步骤 1 解压驱动源码包。

将驱动源码包 WS73V100.tar.gz 置于服务器并进行解压，参考“2.3 编译 SDK”配置编译工具链、内核路径、CPU 架构等，命令如下：

```
$ tar -xzf WS73V100.tar.gz
$ cd WS73V100
```

步骤 2 将编译驱动使用的 config 中的宏 CONFIG_INI_BLE_DISABLE_LL_PRIVACY 打开，即

```
CONFIG_INI_BLE_DISABLE_LL_PRIVACY=y
```

步骤 3 完成配置后，执行 make android 命令编译驱动文件，在 output 生成目标文件：

```
$ make android
```

编译结果输出到"output"目录下，如表 3-3 所示。

表3-3 全量编译结果

文件名	说明
plat_soc.ko	WS73 平台驱动模块。
wifi_soc.ko	WS73WiFi 驱动模块。
ble_soc.ko	WS73 蓝牙驱动模块。
sle_soc.ko	WS73 星闪驱动模块。
ws73_cfg.ini	WS73 客制化的配置文件。

----结束

3.4.2 libbt-vendor.so 编译

说明

目前 WS73 使用的是原生的 bluedroid 代码，用户只需要实现 bt vendor 部分即可。

libbt-vendor.so 的编译步骤如下：

步骤 1 进入到 Android 源码根目录，执行如下命令：

```
$ source build/envsetup.sh
$ lunch {product}
```

步骤 2 下载压缩文件并放至 SDK 相应目录下。

将 WS73V100.tar.gz 中解压出来的 libbt-vendor-soc 文件夹放到 android 源码目录下。

```
$ mkdir -p vendor/{vendor}/bt_soc/bt-vendor
$ cp -rf WS73V100R001C00SPC.tar.gz/driver/bsle/vendor/android/libbt-vendor-soc/*
vendor/{vendor}/bt_soc/bt-vendor/
$ cd vendor/{vendor}/bt_soc/bt-vendor
$ mm -j8
```

----结束

按照上述步骤编译完成后，生成的 **libbt-vendor.so** 文件，一般在 out/target/product/{product}/vendor/lib 或 out/target/product/{product}/vendor/lib64 目录下。

说明

- 编译工具链接需要提前配置环境变量，例如 export PATH=$PATH:/home/AIoT/share/arm-himix100-linux/bin，路径根据实际情况配置。
- --prefix=/vendor 目录是对应工具库所安装的位置，dbus 运行依赖/vendor/share/dbus-1 该位置下的配置文件，可根据系统版本情况配置。

3.4.3 bluedroid 经典蓝牙屏蔽

由于 WS73 不支持经典蓝牙，需要屏蔽 bluedroid 中与经典蓝牙相关的部分。

屏蔽的具体内容请请联系技术支持人员获取。

3.4.4 文件部署

根据平台上所使用的路径进行文件存放即可。一般情况为：

- Android7.0 及以下版本使用的是/system 路径。
- Android8.0 及以上版本使用的是/vendor 路径。

3.4.4.1 Android7.0 及以下版本

- 驱动 ko 的路径会存放在/system/lib/modules 或/system/lib64/modules 路径下。
- firmware 文件会存放在/system/etc/firmware 路径下。此路径与平台 uboot 中 firmware_class.path 的设定有关，实际需要根据主控平台的路径确定。
- ini 配置文件会存放在/system/etc/wifi 路径下。
- so 库文件会存放在/system/lib 或/system/lib64 路径下。

3.4.4.2 Android8.0 及以上版本

- 驱动 ko 的路径会存放在/vendor/lib/modules 或/vendor/lib64/modules 路径下。
- firmware 文件会存放在/vendor/firmware 路径下。此路径与平台 uboot 中 firmware_class.path 的设定有关，实际需要根据主控平台的路径确定。
- ini 配置文件会存放在/vendor/etc/wifi 路径下。
- so 库文件会存放在/vendor/lib 或/vendor/lib64 路径下

3.4.5 蓝牙业务调试

配置单板启动 bluetooth 的步骤如下。

步骤 1 将驱动文件拷贝到单板指定目录下。

1. 将 plat_soc.ko 与 ble_soc.ko 拷贝到单板/vendor/lib/modules 目录下。
2. 单板上的 firmware 和 ws73_cfg.ini 的路径需要在 WS73 SDK 中 build/config/*.config 中定义，具体宏为
   firmware:
   CONFIG_FIRMWARE_BIN_PATH=""
   CONFIG_FIRMWARE_WIFICALI_PATH=""
   CONFIG_FIRMWARE_BSLECALI_PATH=""
   CONFIG_FIRMWARE_WOW_PATH=""
   ws73_cfg.ini:
   CONFIG_INI_FILE_PATH=""
3. WS73E 芯片是将 firmware/e 路径下的 bin 文件拷到 firmware 宏配置的单板路径下，非 WS73E 将 firmware/us 路径下的 bin 文件拷到 firmware 宏配置的单板路径下。
4. 将“output/bin/ws73_cfg.ini”文件拷贝到 CONFIG_INI_FILE_PATH 配置的单板目录下。

步骤 2 加载蓝牙驱动文件，命令如下：

```
insmod  /vendor/lib/modules/plat_soc.ko
insmod  /vendor/lib/modules/ble_soc.ko
```

步骤 3 打开 bluetooth，完成上述步骤后，在 UI 界面上找到蓝牙选项，能正常打开蓝牙并扫描到设备，则表示蓝牙配置成功。

步骤 4 Android 原生流程配置。

步骤 1~步骤 3 执行成功后，则说明 bluetooth 驱动和协议栈启动及功能正常。

1. 根据主控平台，把步骤 1 中的驱动文件在 Android 编译时，拷贝到步骤 1 中对应的路径下。
2. 根据主控平台移植流程操作，加载 ko；一般可把步骤 2 操作写到“device/{vendor}/{product}/init.{product}.rc”中。
3. “在 device/{vendor}/{product}/device.mk”中加入如下 bluetooth 依赖的库文件：

```
PRODUCT_PACKAGES += \
libbt_vendor \
libpf_customize_soc
```

4. 根据 Android 系统，编译出镜像，再通过 APP 打开蓝牙即可。

----结束

3.4.6 星闪驱动编译加载

步骤 1 解压驱动源码包

将驱动源码包 WS73V100.tar.gz 置于服务器并进行解压，命令如下：

```
$ tar -xzf WS73V100.tar.gz
$ cd WS73V100
```

步骤 2 参考“2.3 编译 SDK”配置编译工具链、内核路径、CPU 架构，配置 bin 文件、ini 文件存放路径，参考表 3-4。

表3-4 参数配置说明

参数	说明	示例
CONFIG_FIRMWARE_BIN_PATH	配置 ws73.bin 文件路径	CONFIG_FIRMWARE_BIN_PATH="/vendor/bin"
CONFIG_FIRMWARE_WIFICALI_PATH	配置 wifi_cali.bin 文件路径	CONFIG_FIRMWARE_WIFICALI_PATH="/vendor/bin"
CONFIG_FIRMWARE_BSLECALI_PATH	配置 btc_cali.bin 文件路径	CONFIG_FIRMWARE_BSLECALI_PATH="/vendor/bin"
CONFIG_FIRMWARE_WOW_PATH	配置 wow.bin 文件路径	CONFIG_FIRMWARE_WOW_PATH="/vendor/bin"
CONFIG_INI_FILE_PATH	配置 ws73_cfg.ini 文件路径	CONFIG_INI_FILE_PATH="/vendor/bin"
CONFIG_PLAT_DFR_OUTPUT_PATH	配置 ws73 panic 文件存在路径	CONFIG_PLAT_DFR_OUTPUT_PATH="/data/ws73"

步骤 3 配置 ws73_default.config 文件后，执行命令“make android”，编译 WS73 驱动，生成目录文件路径 output/bin，相关目录文件参考表 3-5。

表3-5 目标文件说明

文件名	说明
plat_soc.ko	WS73 平台驱动模块。

文件名	说明
wifi_soc.ko	WS73WiFi 驱动模块。
ble_soc.ko	WS73 蓝牙驱动模块。
sle_soc.ko	WS73 星闪驱动模块。
ws73_cfg.ini	WS73 客制化的配置文件。

步骤 4 将目标文件上传到单板文件系统中，参考表 3-6。

表3-6 目标文件存放路径说明

目标文件	目标文件生成目录	目标文档存放目录
*.ko	output/bin	/vendor/lib/modules
*.bin	firmware/e	/vendor/bin （与步骤 2 中配置路径保持一致）
*.ini	output/bin	/vendor/bin （与步骤 2 中配置路径保持一致）

说明

SDK 中 firmware 具体差异可参考《NW117V100 系列 Wi-Fi、BLE 和 SLE Combo 芯片 用户指南》 产品概述小节。

步骤 5 加载星闪驱动，命令如下：

```
insmod /vendor/lib/modules/plat_soc.ko
insmod /vendor/lib/modules/sle_soc.ko &
```

----结束

深圳市安信可科技有限公司

4 常见问题

4.1 USB 插入单板后未显示模组的 vid 和 pid
4.2 驱动加载出错
4.3 wpa_supplicant 启动失败

4.1 USB 插入单板后未显示模组的 vid 和 pid

问题描述

当 USB 模组插入单板后，Wi-Fi 打不开时，检查到未显示模组的 vid pid。

解决方案

在 USB 模组未插入单板之前，可执行 busybox lsusb 或 lsusb 命令，并记录当前显示出来的 vid pid；再将 USB 模组插入单板，执行 busybox lsusb 或 lsusb 命令，查看是否会显示模组的 vid pid。

如果有显示，则识别正常；如果仍未显示，则需要相关硬件人员查看 USB 模组是否存在异常。

4.2 驱动加载出错

问题描述

在加载驱动时，出现如下错误：

深圳市安信可科技有限公司

```
Unknown symbol cfg80211_inform_bss_frame_data (err 0)
Unknown symbol cfg80211_sched_scan_results (err 0)
Unknown symbol cfg80211_scan_done (err 0)
Unknown symbol cfg80211_sched_scan_stopped (err 0)
```

问题原因

加载 wifi_soc.ko 时，找不到 Linux 内核 CFG80211 相关的符号引用。

解决方案

1. 主控板 Linux 内核中，CFG80211 没有打开，参考“图 4-1”配置 CFG80211 选项，并将重新编译后的内核烧录进主控板。
2. CFG80211 配置为'M'，则需要将主控 Linux 内核目录下的 cfg80211.ko 拷贝至主控板，加载 cfg80211.ko 后，重新加载驱动。

图4-1 CFG80211 内核配置

```
--- Wireless
<*>   cfg80211 - wireless configuration API
[ ]     nl80211 testmode command
[ ]     enable developer warnings
[ ]     cfg80211 certification onus
[*]     enable powersave by default
[ ]     cfg80211 DebugFS entries
[*]     use statically compiled regulatory rules database
[ ]       support CRDA
[*]     cfg80211 wireless extensions compatibility
<*>   Generic IEEE 802.11 Networking Stack (mac80211)
[*]   Minstrel
[*]     Minstrel 802.11n support
[ ]       Minstrel 802.11ac support
      Default rate control algorithm (Minstrel)  --->
[ ]   Enable mac80211 mesh networking (pre-802.11s) support
[ ]   Export mac80211 internals in DebugFS
[ ]   Trace all mac80211 debug messages
[ ]   Select mac80211 debugging features  ----
```

4.3 wpa_supplicant 启动失败

问题描述

加载驱动完成后，启动 wpa_supplicant，但是启动失败。

解决办法

1. 为“wpa_supplicant”添加可执行权限。
2. 执行“wpa_supplicant”命令时，确定指定路径下存在 wpa_supplicant.conf 配置文件，且配置文件正确。
3. “wpa_supplicant.conf”文件中会记录 wlan0 存放位置，确保该目录下具有读写权限。

深圳市安信可科技有限公司