User’s Guide

WiLink8 Linux Wi-Fi 驱动程序 R8.8 版用户指南

摘要

WiLink8™ Wi-Fi® 支持 IEEE802.11 标准和专有增强功能。本文档可作为用户指南，指导用户如何将 R8.8 Wi-Fi 驱动程序版本及关联的软件组件集成到基于主机 Linux® 的平台。本文档详细介绍了 R8.8 版本的驱动程序架构、核心组件、配置文件、构建过程和测试。本文档还用作验证基本 Wi-Fi 功能的用户指南，并概述了调试和常见问题解答。提供的示例采用 PROCESSOR-SDK-LINUX-AM335X 06_00_00_07。但同样的方法也适用于其他基于 Linux 内核版本 4.19.38 (2019 LTS) 的 SDK。

内容

商标

1 驱动程序支持的功能

下表列出了 WiLink8 驱动程序和器件支持的功能。

- Linux 开源 Wi-Fi 软件包。
- TI NLCP 版本通过了 Wi-Fi Alliance 预认证。
- IEEE：802.11 a/b/g/n、2X2 MIMO (2.4GHz) 和天线分集 (5GHz)
- 支持的模式：STA、AP、P2P、Wi-Fi Direct、Wi-Fi 网状网络
- 吞吐量高达 100 Mbps (UDP)
- 安全性：WPA3、WMM-PS、WMM-AC、WPA/2PSK、Ent、WPS 和 WPSv2
- 低功耗支持：基站 WoW 与暂停/恢复、AP ELP (800 µA 空闲连接)
- 与其他 2.4GHz 协议共存：BT/BLE 和 TI ZigBee (2.4GHz)
- AP DFS，雷达检测 (5 GHz)
- 多角色、多通道：在单个器件上并发运行 2 个 WLAN 角色。
- Wi-Fi 网状网络支持：开放式 802.11s

如需了解 R8.8 中更新的具体功能和错误修复，请参阅 WiLink8 R8.8 版本说明。此外，*WiLink™ 8 WLAN 特性用户指南*详细介绍了 WiLink8 支持的各种功能。

2 WL18xx Linux 驱动程序架构概述

WL18xx Linux 驱动程序使用开源组件以及接口驱动程序来使该器件实现 Wi-Fi 功能。图 2-1 概括显示了驱动程序分区和架构。

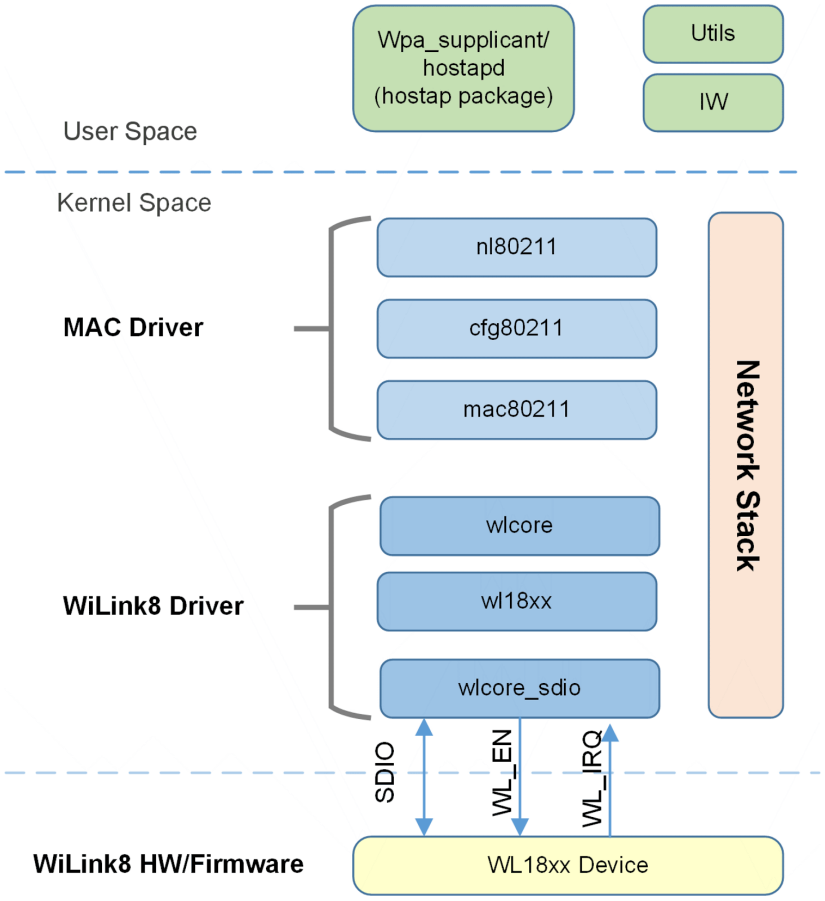

图 2-1. WiLink8 驱动程序架构

以下部分简要介绍了各驱动程序层中的高级组件及其功能。

- WiLink8 固件 – 该固件在器件硬件上运行，可提供 Wi-Fi 的 PHY 和 MAC 功能。主机通过 SDIO 与 WLAN 设备进行通信。在设备侧，WLAN MAC 负责 802.11 MAC 功能，并在外部主机与固件之间传输 WLAN 数据包。MAC 仅负责时序和时间关键型决策。PHY 执行编码/解码和调制/解调的 802.11 PHY 功能，并负责上下调制至载波频率、滤波和放大的射频功能。
- WiLink 驱动程序是器件硬件和固件的抽象层。实现支持 MAC 驱动程序所需的低级操作。
  - wlcore：实现 WiLink 器件的低级驱动程序，可支持 mac80211 操作。包含所有受支持 WiLink™ 芯片组的常见功能。
  - Wl18xx：实现特定于芯片的功能和服务。通过实现特定于硬件的功能来支持 wlcore。
  - wlcore_sdio：SDIO 驱动程序和 WiLink 驱动程序之间的适应层。
- MAC 驱动程序实现 2 层 Wi-Fi 协议要求（数据和控制路径）。这是一个通用组件，而不是特定于平台/器件。这一层包含以下组件。
  - nl80211：在用户空间与 Linux 无线解决方案的内核空间组件之间实现 netlink 接口。
  - cfg80211：Linux 无线配置 API。（该层为软 MAC 和硬 MAC 共用的最底层。）
  - mac80211：为 Wi-Fi 软 MAC 解决方案实现 MAC 层功能的 Linux 内核模块。
- Hostap 软件包：包含开源用户空间软件包。为所有 WLAN 角色（STA、AP、P2P 和网状网络）提供上层管理层。生成 2 个守护程序：wpa_supplicant（STA、P2P、网状网络）和 hostapd (AP)。
- 实用程序提供初始化和配置服务。实现调试和统计功能。

3 平台集成

以下部分详细介绍了如何将驱动程序集成到 Linux SDK 平台中。提供的参考和指令适用于任何采用 Linux 操作系统的平台。不过，下文提到的具体指令基于 PROCESSOR-SDK-LINUX-AM335X 06_00_00_07。对于 WiLink8 硬件集成，请参阅 *WiLink™ 模块硬件集成指南*。

下面提供了手动将 WiLink8 R8.8 驱动程序集成到任意内核的一般步骤。同样的方法也可用于将现有 SDK 的 WiLink8 驱动程序版本升级到 R8.8 版。不过，如果与 TI SDK 搭配使用（利用预构建的内核），则可以使用“构建实用程序”脚本。请注意，以下步骤假定 Linux 主机环境已启动并且正在运行。有关设置 Linux 主机 PC 的更多信息，请遵循 Processor SDK 入门指南中提供的说明。

1. 下载内核 (4.19+) 和平台 SDK。
2. 根据 SDK 安装说明，将 SDK 映像安装到 SD 卡中。
3. 使用 verify_kernel.sh 实用程序或手动配置内核
4. 应用内核补丁 – 如果是首次构建内核，则需要完成此步骤。
5. 构建 WLAN 模块、内核 zImage（可选）、内核模块 (构建实用程序) 和 BeagleBone Black DTB。
6. 为特定电路板编译器件树文件 (dts → dtb) 并进行更新。
7. 将构建输出复制到 SD 卡上。

第 1 步和第 2 步需要手动完成。无论使用何种 SDK，首先都要完成这些步骤。这些步骤将确保将 SDK 提供的默认操作系统安装到 SD 卡中。R8.8 WiLink8 WLAN 驱动程序版本中只会构建与 WLAN、内核和模块相关的一部分组件，因此务必要安装整个默认文件系统。

第 3 步、第 4 步和第 6 步（第 5 步“DTS/B 文件”除外）可以使用“build_scripts”实用程序来执行。DTS/B 文件特定于硬件，并需要根据硬件设计进行创建。

3.1 电路板器件树所需的配置 (DTS/DTB)

器件树是一系列用于描述硬件平台的 Linux 源代码文件。若要集成 WilInk8，该文件需要具有合适的配置，以便与所需的硬件接口相集成。对于 WiLink8 WLAN 功能，这类文件包括：

- SDIO 配置
- WL_EN GPIO 配置
- WL_IRQ GPIO 配置
- 电源控制配置
- MMC 配置

默认 TI Processor SDK dts/b 文件已经启用了所需的配置。系统会在启动时根据所用的硬件平台自动选择此类文件。有关所需配置的详细信息，请参阅 AM335x_evm.dts。请注意，可能需要根据所用的具体硬件平台来修改这些设置。

3.2 针对 TI WLAN 驱动程序配置内核

大多数内核配置都是默认的，并已在 TI SDK 版本中进行了配置。以下步骤旨在确保在 .config 文件中选择了正确的内核配置并在构建内核时加以应用。

以下命令将会打开 makemenu config 以手动启用所需的配置。请注意，使用“build utilities”脚本期间，系统会自动启用所需的配置。

```
export PATH=<sdk path>/linux-devkit/sysroots/x86_64-arago-linux/usr/bin:$PATH
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- distclean
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- <config file>
make ARCH=arm CROSS_COMPILE=arm-linux-gnueabihf- menuconfig
```

需要为 WiLink8 驱动程序启用以下内核配置标志/开关。

```
CONFIG_CFG80211=m
CONFIG_MAC80211=m
CONFIG_WLCORE=m
CONFIG_WLCORE_SDIO=m
CONFIG_WL18XX=m
CONFIG_NL80211_TESTMODE=y
CONFIG_MAC80211_MESH=y
```

所需的其他配置均包含在 *verify_kernel_config.sh*（后续部分中会详细介绍）中，后者可用于更新内核配置。

3.3 电路板器件树所需的配置 (DTS/DTB)

器件树是一系列用于描述硬件平台的 Linux 源代码文件。若要集成 WilInk8，该文件需要具有合适的配置，以便与所需的硬件接口相集成。对于 WiLink8 WLAN 功能，这类文件包括：

- SDIO 配置
- WL_EN GPIO 配置
- WL_IRQ GPIO 配置
- 电源控制配置
- MMC 配置

默认 TI Processor SDK dts/b 文件已经启用了所需的配置。系统会在启动时根据所用的硬件平台自动选择此类文件。有关所需配置的详细信息，请参阅 AM335x_evm.dts。请注意，可能需要根据所用的具体硬件平台来修改这些设置。

3.4 使用构建实用程序构建 R8.8 版本

以下部分详细介绍了使用构建实用程序构建 R8.8 版本的步骤。构建实用程序提供了统一的方法来构建、更新和集成 WL18xx WLAN 驱动程序和模块。此步骤需要在针对目标平台配置内核并对 DTS/B 文件进行必要更改后完成。**WiLink8 R8.8 版本基于 Linux 内核版本 4.19，并且不支持之前内核版本的反向端口**。

构建实用程序集成了操作 WL18xx 器件所需的 WiLink8 WLAN 模块以及驱动程序。它还包含用于 WPA 请求的额外构建软件包和基于开源但针对 WL18xx 器件定制的 hostapd。实用程序还集成了用于测试的工具、用于演示 Wi-Fi 操作的示例脚本以及器件固件。脚本可用于构建整个模块和内核，或者能够构建各个模块。

构建实用程序脚本中所含的一般过程如下所示：

1. 下载构建实用程序软件包
2. 配置设置
3. 克隆适用于特定版本 (R8.8) 的源代码和驱动程序组件
4. 使用 verify_kernel.sh 添加适当的内核配置
5. 应用内核补丁 (只需一次)
6. 构建所有版本的二进制文件或各个组件
7. 将生成的二进制文件安装至目标文件系统

脚本将下载相关组件的以下源文件。如需了解各个组件确切版本的详细信息，请参阅 WiLink8 R8.8 版本说明。下载的源文件保存在 ./build-utilities/src 目录中。

目录	内容
fw_download	包含版本随附的 WiLink8 器件固件
hostap	wpa_supplicant 和 hostapd 的源代码。它们依赖于 opnessl 和 libnl，后两者也会下载到 ./build-utilities/src 下
iw	iw 工具的源代码。
openssl	包含克隆的 openssl 源代码
scripts_download	用于运行 WL18xx 的各种脚本
ti_utils	TI 提供的各种实用程序。
wireless-regdb	无线管制数据库

1. **下载构建脚本并切换至 R8.8 分支。**

   从 git://git.ti.com/wilink8-wlan/build-utilites.git 克隆构建实用程序，示例如下：

```
user@ubuntu:~/ti-sdk-am335x-evm-07.00.00.00$ cd ~/wl8-build/
user@ubuntu:~/wl8-build$ git clone git://git.ti.com/wilink8-wlan/build-utilites.git
Cloning into 'build-utilites'...
remote: Counting objects: 888, done.
remote: Compressing objects: 100% (412/412), done.
Recremote: Total 888 (delta 490), reused 761 (delta 456)
Receiving objects: 100% (888/888), 12.82 MiB | 5.10 MiB/s, done.
Resolving deltas: 100% (490/490), done.
user@ubuntu:~/wl8-build$ cd build-utilites/
user@ubuntu:~/wl8-build/build-utilites$ ls
build_wl18xx.sh  configuration/  configuration.sh  patches/  README  setup-env.sample
sudo_build_wl18xx.sh  verify_kernel_config.sh
```

   使用以下命令切换至 R8.8 分支：

```
user@ubuntu:~/wl8-build/'build-utilites'$ git checkout r8.8
```

克隆完成后，构建实用程序的以下脚本会出现在 ./build-utilities 文件夹中。下面提供了相关脚本的详细信息：

setup-env.sample	示例环境设置文件。应复制并重命名为 setup-env
configuration.sh	包含用于下载源文件的 Git 存储库地址和 Git 标签的配置详细信息。
build_wl18xx.sh	这是主脚本，该脚本会使用 setup-env 和 configuration.sh 以及各个用户参数来下载、清理、更新或构建用户选择的特定组件。
sudo_build_wl18xx.sh	与使用 sudo 选项的 build_wl18xx.sh 相同。请注意，如果将该脚本的“sudo”版本用于“init”选项，相关目录会变为归 root 所用。
verify_kernel_config.sh	用于验证内核配置的脚本。

以下各节将使用 build-wl18xx.sh 脚本来清理、构建和安装所有组件或特定组件。构建特定组件的情况将在稍后部分进行讨论。使用 –h 参数则可以显示可用的命令选项。

```
./build_wl18xx.sh -h
```

下面显示了可用的选项。

```
user@ubuntu:~/R8.8/build-utilites$ ./build_wl18xx.sh -h
This script builds all/one of the relevant wl18xx software packages.

Usage :

Building full package : Build all components except kernel, dtb
    ./build_wl18xx.sh init                                      [ Download and Update w/o
build  ]
                      update       R8.8                     [ Update to specific TAG & Build ]
                      clean                                 [ Clean & Build                  ]
                      check_updates                         [ Check for build script updates ]

Building specific component :
                      hostapd                               [ Clean & Build hostapd          ]
                      wpa_supplicant                        [ Clean & Build wpa_supplicant   ]
                      modules                               [ Clean & Build driver modules   ]
                      firmware                              [ Install firmware binary        ]
                      scripts                               [ Install scripts                ]
                      utils                                 [ Clean & Build scripts          ]
                      iw                                    [ Clean & Build iw               ]
                      openssl                               [ Clean & Build openssll         ]
                      libnl                                 [ Clean & Build libnl            ]
                      wireless-regdb                        [ Install wireless regdb         ]
                      patch_kernel                          [ Apply provided kernel patches  ]
                      kernel <defconfig filename>         [ Clean & Build Kernel            ]
                      kernel_noclean <defconfig_filename> [ Build Kernel w/o clean         ]
                      patch_bbbe14_dts [Patch bbb black dts file to add e14 cape support]
```

2. **创建 setup-env 文件。**

“setup-env.sample”文件用作“setup-env”的基础，后者包含特定于用户的环境变量。用户应将 setup-env.sample 复制到 setup-env 并根据特定于用户的环境编辑相关变量。用户应编辑 setup-env 文件，以指向内核和工具链所在的正确目录。下面是一个示例文件：

```
#                           \\\//
#                          -(o o)-
#=======================oOO==(_)==OOo=======================
# This file contains the exports needed for automating the
# build process of WLAN components.
# Place this file in the same directory with wl18xx_build.sh
# build scripts.No need to run 'source setup-env', the build
# scripts will perform it internally.
#============================================================
# User specific environment settings - use full PATH

# TOOLCHAIN_PATH setting is mandatory. ex: TOOLCHAIN_PATH=/opt/ti-processor-sdk-linux-am335x-
evm-06.00.00.07/linux-devkit/sysroots/x86_64-arago-linux/usr/bin
export TOOLCHAIN_PATH=
```

```
# ./fs folder will be created if ROOTFS is set to DEFAULT
export ROOTFS=DEFAULT

# KERNEL_PATH setting is mandatory. ex: KERNEL_PATH=/opt/ti-processor-sdk-linux-am335x-
evm-06.00.00.07/board-support/linux-4.19.38+gitAUTOINC+4dae378bbe-g4dae378bbe
export KERNEL_PATH=

# CROSS_COMPILE setting is mandatory
export CROSS_COMPILE=arm-linux-gnueabihf-

# ARCH setting is mandatory
export ARCH=arm
[ "$TOOLCHAIN_PATH" != "" ] && export PATH=$TOOLCHAIN_PATH:$PATH
```

setup-env 文件应与构建脚本（build_wl18xx.sh 等）位于同一目录下。

3. **下载源（又称“初始化”）库。**

   以下步骤会下载构建所需的整个源代码。首次安装时，这可能需要更长的时间。后续更新时，时间会短一些。

```
user@ubuntu:~/wl8-build$ cd build-utilites
user@ubuntu:~/wl8-build/build-utilites$ ./build_wl18xx.sh init
```

4. **验证并设置所需的内核配置。**

   WiLink8 WLAN 功能要求在内核配置中启用一些设置。这可以使用构建实用程序软件包中提供的 *verify_kernel_config.sh* 实用程序来设置/验证，如下所示：

```
user@ubuntu:~/wl8-build/build-utilites$ ./verify_kernel_config.sh <def_config file>
```

   示例：user@ubuntu:~/wl8-build/build-utilites$ ./verify_kernel_config.sh /opt/ti-processor-sdk-linux-am335x-evm-06.00.00.07/board-support/linux-4.19.38+gitAUTOINC+4dae378bbe-g4dae378bbe/arch/arm/configs/tisdk_am335x-evm_defconfig

5. **应用所需的内核补丁。**

   WiLink8 驱动程序包中含有一系列补丁，需要应用这些补丁，才能启用完整的功能。这些补丁起到了功能增强和错误修复的作用。只有首次构建内核映像来启用 WiLink8 WLAN 时，才需要完成此步骤。

```
user@ubuntu:~/wl8-build/build-utilites$ ./build_wl18xx.sh patch_kernel
```

6. **构建内核。**

   以下命令将会构建内核。若要包含特定于 TI WiLink8 的补丁，需要重建内核。**用户可以直接从 SDK 构建内核，也可以使用构建脚本来构建内核**。内核 defconfig 文件名作为参数传递。

```
user@ubuntu:~/wl8-build/build-utilites$ ./build_wl18xx.sh kernel <defconfig file>
Ex: user@ubuntu:~/wl8-build/build-utilites$ ./build_wl18xx.sh kernel tisdk_am335x-evm_defconfig
```

   使用 BeagleBone Black Cape 和 BeagleBone 进行开发时，需要执行以下步骤。

   备注

   内核映像 zImage 也随着安装文件系统一同构建。

7. **应用补丁并构建 BeagleBone DTB 文件（可选）。**

   将 BeagleBone Black 与 Element-14 无线 Cape 搭配使用时需要用到的 DTS/B 文件也可以使用 build_utilities 来生成。应用 BeagleBone Black dts 文件来使用以下命令并通过 WL1837MOD 添加对 Element-14 无线 Cape 的支持

```
user@ubuntu:~/wl8-build/build-utilites$ ./build_wl18xx.sh patch_bbbe14_dts
```

构建具有 Element 14 无线 Cape 的 Beaglebone Black dts 来生成所需的 dtb 文件。

```
user@ubuntu:~/wl8-build/build-utilites$ ./build_wl18xx.sh bbbe14_dtb
```

8. **构建版本二进制文件。**

   每个 WiLink8 驱动程序版本都带有特定的标签。对于 Wilink8 Driver R8.8 驱动程序，该标签为“**R8.8**”。若要检出 R8.8 版本、进行构建并安装到目标文件系统中，则需要使用以下命令（假定具有访问文件系统所需的根权限）：

```
user@ubuntu:~/wl8-build/build-utilites$ ./build_wl18xx.sh update R8.8
```

9. **安装 WiLink8 版本二进制文件。**

在此阶段，WiLink8 驱动程序和内核的所有组件应该都已构建完毕。输出库、二进制文件、示例脚本、固件和 TI 实用程序等均位于 setup_env 文件中指定的文件夹内（默认为 *./build-utilities/fs)*。这些均可直接安装。请注意，以下步骤假定默认的 SDK 映像已安装到 SD 卡中。使用以下示例命令将 fs 文件夹从 ./build-utilities 目录复制到目标位置。

```
sudo cp -p ./fs/* <rootfs path on SD card>/
```

3.5 分别构建 WiLink8 驱动程序版本二进制文件

构建实用程序还支持清理、构建和安装一个特定的组件，而不是全部组件。可以使用以下命令来查看脚本中提供的各种选项。

```
./sudo_build_wl18xx.sh -h
```

下面显示了可用的选项：

openssl/libnl	清理和构建 openssll 与 libnl。需要这些库才能构建 hostapd/wpa_supplicant/iw 等用户空间组件
hostapd/wpa_supplicant/iw	清理和构建 hostapd、wpa_supplicant 或 iw 用户空间实用程序。在进行此构建前，应先构建 openssl 和 libnl（顺序非常重要）
modules	清理和构建 WiLink8 驱动程序模块
firmware	将固件二进制文件安装到 ./build-utilities/fs
scripts	安装 TI 示例脚本以运行 STA、AP、MESH
utils	清理和构建 wlconf 等脚本
patch_kernel	应用已提供的未上载的内核补丁。
kernel	清理构建内核
kernel_noclean	构建内核而不进行清理。对于增量构建很有用。
patch_bbbe14_dts	让 Beaglebone Black 使用 Element 14 无线 Cape 所需的补丁

构建驱动程序专用组件的命令语法：

```
./sudo_build_wl18xx.sh <module>
```

上述命令假定需要根权限才能安装至文件系统。

4 引导和 WLAN 启动

以下步骤中的硬件设置采用 AM335x EVM 与 WL1837MODCOM8I 模块。SD 卡现在安装在 AM335x EVM 的 SDMMC1 插槽中。使用 UART 电缆将 J12 UART 连接器与 PC 相连。运行任意终端程序（例如：TerraTerm）并将波特率配置为 115200、8 位、无极性。也可以使用 BeagleBone Black 与 Element 14 无线 Cape 来替代以下设置。

图 4-1. 为驱动程序启动采用 AM335x 和 WiLink8 的基本硬件设置

内核引导期间应该会显示以下消息，指示驱动程序已启动且 WiLink8 固件已下载。

```
[   28.358451] wlcore: wl18xx HW: 183x or 180x, PG 2.2 (ROM 0x11)
[   28.478778] wlcore: loaded
[   29.515823] wlcore: PHY firmware version: Rev 8.2.0.0.244
[   29.662595] wlcore: firmware booted (Rev 8.9.0.0.84)
```

另外，登录完成后，默认的 wlan0 接口应该会打开。

```
root@am335x-evm:~# ifconfig wlan0
wlan0     Link encap:Ethernet  HWaddr 0C:1C:57:BB:60:5E
          UP 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)
root@am335x-evm:~#
```

上述步骤确认驱动程序已启动并且正在运行。

4.1 配置 WiLink8 目标

WiLink8 驱动程序随附了默认的 *wl18xx-conf.bin* 二进制文件。*/lib/firmware/ti-connectivity/wl18xx-conf.bin* 会配置所需的射频参数、使用的天线数量，以及所需的工作频段等。此文件需要根据设置中所用的 WL18xx 器件或模块进行调整。为了简化生成 *wl18xx-conf.bin*（*位于 /lib/firmware/ti-connectivity 中*）的过程，R8.8 版本中包含可用于根据硬件配置做出适当选择的配置脚本。以下各节详细介绍了“*configure-device.sh*”脚本的用法。该脚本将使用所选的选项来更新现有的 wlconf 文件。请注意，编程的值会影响射频性能，并且需要准确的值才能通过认证。

configure-device.sh 是一个菜单式脚本。该脚本会询问硬件相关问题，用于正确配置要与 WiLink 8 器件配套使用的目标。此脚本使用 *'wlconf'* 实用程序来创建 WiLink8 配置二进制文件。*WiLink™ 8 解决方案 WiLink8 – wlconf* 包含有关“wlconf”实用程序以及如何修改此配置的更多详细信息。

首先，完成上述启动过程并确保 WiLink8 器件已启动。与目标设备建立串行连接并以 *root* 身份登录。

导航至脚本所在位置 /usr/sbin/wlconf 并运行 *configure-device.sh*：

```
cd /usr/sbin/wlconf
./configure-device.sh
```

该脚本将会向您询问一系列硬件相关问题并针对您的系统配置 wlconf。此脚本可用于 WiLink8 TI 模块、非 TI 模块或板载芯片设计。下面显示了一些示例。

使用 **WiLink8 TI 模块**时，该脚本将会自动从 `/usr/sbin/wlconf/official_inis` 中挑选正确的 INI 文件：

```
Example for WL1837 without Japanese Certification:

root@am335x-evm:/usr/sbin/wlconf# ./configure-device.sh
```

```
Please provide the following information.

Are you using a TI module? [y/n] : y
What is the chip flavor? [1801/1805/1807/1831/1835/1837 or 0 for unknown] : 1837
Should Japanese standards be applied? [y/n] : n
How many 2.4GHz antennas are fitted? [1/2] : 2
How many 5GHz antennas are fitted (using 2 antennas requires a proper switch)? [0/1/2] : 1
[  106.068069] wlcore: down
------------------------------------------------------------

The device has been successfully configured.
TI Module: y
Chip Flavor: 1837
Base INI file used: /usr/sbin/wlconf/official_inis/WL1837MOD_INI_FCC_CE.ini
Number of 2.4GHz Antennas Fitted: 2
Number of 5GHz Antennas Fitted: 1
Diversity Support: y
SISO40 Support: y
Japanese Standards Applied: n

------------------------------------------------------------


Example for WL1801

root@am335x-evm:/usr/sbin/wlconf# ./configure-device.sh

Please provide the following information.

Are you using a TI module? [y/n] : y
What is the chip flavor? [1801/1805/1807/1831/1835/1837 or 0 for unknown] : 1801
Should SISO40 support be applied? [y/n] : y
[  539.778261] wlcore: down

------------------------------------------------------------
The device has been successfully configured.
TI Module: y
Chip Flavor: 1801
Base INI file used: /usr/sbin/wlconf/official_inis/WL1835MOD_INI_C2PC.ini
Number of 2.4GHz Antennas Fitted: 1
Number of 5GHz Antennas Fitted: 0
Diversity Support: n
SISO40 Support: y
Japanese Standards Applied: n
------------------------------------------------------------
```

对于 **WiLink8 板载芯片或非 TI 模块**配置，该脚本将会挑选 /usr/sbin/wlconf/official_inis/WL8_COB_INI.ini。TI 提供了此文件的模板。 **客户将需要根据其设计修改此文件中的射频参数，或者使用各个模块供应商提供的值**。

```
root@am335x-evm:/usr/sbin/wlconf# ./configure-device.sh

Please provide the following information.

Are you using a TI module? [y/n] : n
What is the chip flavor? [1801/1805/1807/1831/1835/1837 or 0 for unknown] : 1837
How many 2.4GHz antennas are fitted? [1/2] : 2
How many 5GHz antennas are fitted (using 2 antennas requires a proper switch)? [0/1/2] : 1
[  148.158965] wlcore: down

------------------------------------------------------

The device has been successfully configured.
TI Module: n
Chip Flavor: 1837
Base INI file used: /usr/sbin/wlconf/official_inis/WL8_COB_INI.ini
Number of 2.4GHz Antennas Fitted: 2
Number of 5GHz Antennas Fitted: 1
Diversity Support: n
SISO40 Support: y
Japanese Standards Applied: n

------------------------------------------------------
```

5 测试基本的 WLAN 功能

以下各节提供了验证 STA、AP、STA+AP（多角色）和网状网络功能构建情况的步骤。在之前的构建流程中已经安装了执行这些测试所需的脚本。所用设置基于 AM335x EVM 以及 TI WL1837MODCOM8I 评估模块。

在继续之前，请依照前文所述的构建流程以确保 SD 卡上存在有效的映像。相关脚本位于 */usr/share/wl18xx/* 中。此文件夹包含以下用于验证功能的示例脚本

```
root@am335x-evm:/usr/share/wl18xx# ls
ap_start.sh           mesh_start.sh            ps_lock.sh               udhcpd.conf
ap_stop.sh            mesh_stop.sh             set_cmd_silence.sh       udhcpd.leases
calibrate.sh          mesh_supplicant.conf     sta_connect-ex-dhcp.sh   udhcpd2.conf
dynamic-debug.sh      mod_start.sh             sta_connect-ex.sh        udhcpd_mesh.conf
entropy.bin           mod_stop.sh              sta_start.sh             unload_wlcore.sh
hostapd.conf          p2p_cli.sh               sta_stop.sh              wlconf-toggle-set.sh
load_wlcore.sh        p2p_start.sh             testing-boot.sh          wlcore-print-fw-stat.sh
mesh_bridge.sh        p2p_stop.sh              testing.ini              wpa_supplicant.conf
mesh_join.sh          print_stat.sh            testing_set_wlcore.sh
root@am335x-evm:/usr/share/wl18xx#
```

请注意，此文件夹中还包含 wpa_supplicant 和 hostapd 所需的配置文件。您需要修改这些文件来启用 WPA2、通道选择和工作模式等。

5.1 STA 模式

以下部分详细介绍了如何将 WiLink8 器件置于 STA 模式并连接到接入点，以及验证该基站和 AP 之间的连接性。这里使用的硬件设置与前文所述相同。

图 5-1. Wi-Fi 基站硬件设置

使用基站模式预构建脚本的一般流程如下：

- 导航至包含开箱即用脚本的目录
- 启动基站模式
- 连接到非安全接入点
- 从接入点请求 IP 地址
- 对接入点执行 Ping 操作以验证连接

5.1.1 非安全 AP 的基站模式流程

```
root@am335x-evm:~# cd /usr/share/wl18xx/
root@am335x-evm:/usr/share/wl18xx# ./sta_start.sh
root@am335x-evm:/usr/share/wl18xx# Successfully initialized wpa_supplicant

root@am335x-evm:/usr/share/wl18xx# ./sta_connect-ex.sh <SSID Ex:abc>
root@am335x-evm:/usr/share/wl18xx# udhcpc -i wlan0
```

5.1.2 安全 AP 的基站模式流程

```
root@am335x-evm:~# cd /usr/share/wl18xx/
root@am335x-evm:/usr/share/wl18xx# ./sta_start.sh
root@am335x-evm:/usr/share/wl18xx# Successfully initialized wpa_supplicant

root@am335x-evm:/usr/share/wl18xx# ./sta_connect-ex.sh <SSID Ex:abc> WPA-PSK <password Ex:12345678>
root@am335x-evm:/usr/share/wl18xx# udhcpc -i wlan0
```

5.1.3 确认连接

若要确认连接，请使用 ping 命令。例如，如果接入点 IP 地址为 192.168.1.1，我们将在 EVM 上调用以下命令：

```
ping 192.168.1.1
```

您应该会看到类似于以下内容的输出：

```
PING 192.168.1.1 (192.168.1.1): 56 data bytes
64 bytes from 192.168.1.1: seq=0 ttl=64 time=1003.369 ms
64 bytes from 192.168.1.1: seq=1 ttl=64 time=2.526 ms
```

您也可以调用该命令。此命令将显示 AP 设置并验证连接 `iw wlan0 link`。

5.2 AP 模式

以下各节详细介绍了如何将 WiLink8 器件置于 AP 模式并连接其他 Wi-Fi 设备，以及验证所连设备和 AP 之间的连接性。这里使用的硬件设置与前文所述相同，只是增加了一个 Wi-Fi 设备。

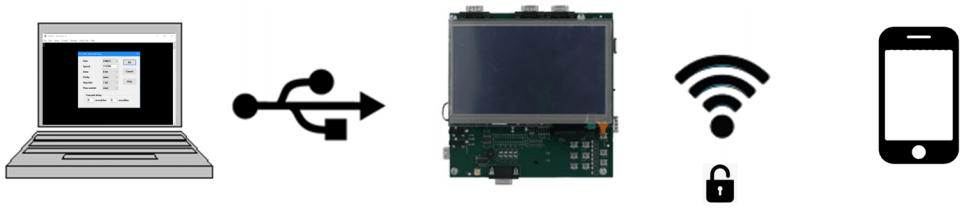

图 5-2. Wi-Fi 接入点 (AP) 硬件设置

5.2.1 AP 模式流程

通过编辑 /usr/share/wl18xx/hostapd.conf 中的 hostapd.conf 文件，配置 AP。这个文件中提供了一些选项，可用于更改 SSID、安全性以及文件中所示的其他高级功能。下面列出了要考虑的重要参数：

```
# SSID to be used in IEEE 802.11 management frames
ssid=xyzabc

# Channel number (IEEE 802.11)
channel=6

# ieee80211n: Whether IEEE 802.11n (HT) is enabled
# 0 = disabled (default)
# 1 = enabled
# Note: You will also need to enable WMM for full HT functionality.
ieee80211n=1

# ht_capab: HT capabilities (list of flags)
ht_capab=[SHORT-GI-20][GF][HT]

##### WPA/IEEE 802.11i configuration ##########################################
wpa=2
wpa_passphrase=wilink80

# Set of accepted key management algorithms (WPA-PSK, WPA-EAP, or both).The
# entries are separated with a space.WPA-PSK-SHA256 and WPA-EAP-SHA256 can be
# added to enable SHA256-based stronger algorithms.
# (dot11RSNAConfigAuthenticationSuitesTable)
wpa_key_mgmt=WPA-PSK

# Operation mode (a = IEEE 802.11a, b = IEEE 802.11b, g = IEEE 802.11g,
# Default: IEEE 802.11b
hw_mode=g

# Pairwise cipher for WPA (v1) (default: TKIP)
wpa_pairwise=TKIP CCMP

# Pairwise cipher for RSN/WPA2 (default: use wpa_pairwise value)
rsn_pairwise=CCMP
```

5.2.2 启动 AP

运行 ap_start.sh 脚本：

```
root@am335x-evm:/usr/share/wl18xx# ./ap_start.sh
adding wlan1 interface
Configuration file: /usr/share/wl18xx/hostapd.conf
[ 4398.173284] IPv6: ADDRCONF(NETDEV_UP): wlan1: link is not ready
wlan1: interface state UNINITIALIZED->COUNTRY_UPDATE

root@am335x-evm:/usr/share/wl18xx# Using interface wlan1 with hwaddr 0c:1c:57:bb:60:5f and ssid
"kns"
[ 4403.305047] netlink: 'hostapd': attribute type 213 has an invalid length.
[ 4403.321775] IPv6: ADDRCONF(NETDEV_CHANGE): wlan1: link becomes ready
[ 4403.430771] cryptd: max_cpu_qlen set to 1000
wlan1: interface state COUNTRY_UPDATE->ENABLED
wlan1: AP-ENABLED
root@am335x-evm:/usr/share/wl18xx#
```

5.2.3 确认连接

若要验证 AP 广播，请使用任意商用基站（智能手机、笔记本电脑等等）进行连接。当基站成功连接到 WiLink AP 时，您应该会看到一个提示 (AP_STA_CONNECTED)。

5.3 多角色（AP +STA 模式）

此演示展示了如何利用 /usr/share/wl18xx 中开箱即用的脚本来将 WiLink 器件同时用作基站和 AP。多角色 Wi-Fi 器件的一个示范用例可能是作为无线网桥使用。在多角色模式中，WiLink 器件可以作为客户端连接到支持互联网的安全 AP（STA 模式），同时用作基站，其他设备可以连接到该基站进行互联网访问。这时可以使用用于 AP 模式测试的相同硬件。

5.3.1 多角色连接的一般流程

1. 启动基站角色。
2. 连接至接入点。
3. 启动 AP 角色。

需要运行以下脚本以启用 STA 和 AP 角色。AP 角色中提及的 hostapd.conf 更新也适用于此流程。

```
./sta_start.sh
./sta_connect-ex.sh exampleSSID WPA-PSK examplepassword
udhcpc -i wlan0
./ap_start.sh
```

如果未更改默认的 hostapd.conf 设置，AP 现在应该会以 SitaraAP 身份进行广播，同时也会以基站身份连接至“启动基站角色”部分中指定的 AP。若要验证基站连接，请调用：

```
iw wlan0 link
```

此时应该会显示所连 AP 的详细信息。若要验证 AP 广播，请使用任意商用基站（智能手机、笔记本电脑等）进行连接。当基站成功连接到 WiLink AP 时，您应该会看到一个提示 (AP_STA_CONNECTED)。

5.4 IEEE802.11s 网状网络模式

WiLink8 器件也支持 IEEE 802.11s 网状网络功能。若要验证此模式，最少需要两个前文使用的 WiLink8 设置。还可以将更多的器件添加至该网状网络。

图 5-3. Wi-Fi 网状网络硬件设置

如需了解网状网络、受支持拓扑结构以及如何使用示例脚本来启动该网络的更多详情，请参阅 *WiLink™ 8 WLAN 软件 - 802.11s 网状网络*。

6 参考文献

- 德州仪器 (TI)：*WiLink™ 8 WLAN 特性用户指南*
- 德州仪器 (TI)：*WiLink™ 模块硬件集成指南*
- 德州仪器 (TI)：*WiLink™ 8 解决方案 WiLink8 – wlconf*
- 德州仪器 (TI)：*WL18xx .ini 文件*
- 德州仪器 (TI)：*WL18x7MOD WiLink™ 8 双频带工业模块 – Wi-Fi®、Bluetooth® 和低功耗 Bluetooth® (LE) 数据表*

A 常见问题解答和调试提示

以下部分旨在提供在 Linux 平台上集成和运行 WiLink8 驱动程序时遇到的一般问题。如需查看更全面的常见问题解答列表和其他帮助信息，可以访问 E2E 论坛。

问：如何判断 Wi-Fi 功能是否正常？

答：打开 WLAN 接口并使用“iw”实用程序执行扫描：

```
ifconfig wlan0 up
```

此时应该会显示以下消息：

```
wlcore: PHY firmware version: Rev 8.2.0.0.244
wlcore: firmware booted (Rev 8.9.0.0.84)
```

接下来，执行扫描并查看扫描结果：

```
iw wlan0 scan | grep <SSID>
        SSID: IOP_035
        SSID: Demo_24
        SSID: externalhotspot84
```

如果遇到了任何错误，请按照以下步骤操作：

1. 检查器件配置。

   您记得在初始化时使用 configure-device.sh 脚本了吗？

   备注

   该脚本位于 /usr/share/wl18xx/configure-device.sh 中。

   确保您使用的是与您需求匹配的适用 .ini 文件。

   更多信息，请参阅 *WL18xx .ini 文件*。
2. 尝试在 1-2 个其他平台上重现问题。
3. 尝试在使用最新固件和软件驱动程序版本（当前为 R8.8）的情况下重现问题。
4. 尝试在禁用增强型低功耗 (ELP) 模式的情况下重现问题。

   若要禁用 ELP，请执行以下命令：

   ```
   iw wlan0 set power_save off
   echo 0 > /sys/kernel/debug/ieee80211/phy0/wlcore/sleep_auth
   ```
5. 尝试在使用另一同类供应商的情况下重现问题。

   对于 STATION/CLIENT 模式 - 尝试使用其他接入点供应商。

   对于接入点 (AP) 模式 - 尝试使用其他基站供应商。

   对于对等 (P2P) 模式 - 尝试使用其他 P2P 供应商。

   如果问题仍未解决，请在下方查找您的具体用例。

问：我可以使用 *ifconfig* 打开接口，但在执行扫描时，我看到了驱动程序崩溃日志。

答：确保从 wl18xx 器件收到了中断。这可以使用以下命令来实现：

```
cat /proc/interrupts | grep wl18xx
```

接下来应该会看到以下输出或类似内容：

```
54:         15  44e07000.gpio  27 Edge      wl18xx
```

如果 WL_IRQ 引脚配置正确，您应该会看到一个大于“0”的数字，如上所示。如果值为零，请重新访问电路板器件树文件并确保对 WL_IRQ GPIO 进行正确的多路复用，且加载“wlcore_sdio”模块时未看到任何错误。

问：我已经确认连接了正确的引脚并进行了多路复用，但 WLAN 接口仍无法正常工作。

答：确保根据数据表中的设计指导原则遵循了相应的上电和复位序列。更多信息，请参阅 *WL18x7MOD WiLink™ 8 双频带工业模块 – Wi-Fi®、Bluetooth® 和低功耗 Bluetooth® (LE) 数据表* 中的*上电和关断状态*。也就是说，在启动 WLAN_EN 之前，VBAT/VIO 电压和慢时钟 (32kHz) 必须保持稳定。当 WL_IRQ 读取为逻辑“0”时，该模块处于唤醒状态。首次触发 IRQ 时，主机开始通过 SDIO 接口进行通信。

问：我确认遵循了上电和复位序列，但 WLAN 接口仍无法正常工作。

答：确保器件枚举期间检测到了 WLAN 卡。如果平台集成是根据硬件规格指南/平台集成指南完成的，则内核启动期间应能检测到 SDIO 器件。请审查内核引导日志并查看是否有以下消息：

```
[   18.538564] mmc1: new high speed SDIO card at address 0001
```

问：我确认在 SDIO 接口上检测到了 WLAN 器件，但 WLAN 仍无法正常工作。

答：确保使用 WL18xx 处理器维基网页中的 WL8 软件构建流程加载了 WLAN 驱动程序，或者在内核中构建了 WLAN 驱动程序（如果内核版本 >= 4.1）

使用“lsmod”命令时，您应该会看到加载了以下模块：

```
Module                  Size Used by
…
wl18xx                 83954  0
wlcore                186624  1 wl18xx
mac80211              479316  2 wl18xx,wlcore
cfg80211              397999  3 mac80211,wl18xx,wlcore
wlcore_sdio             7829  0
```

问：没错，我确实构建了相关模块，但在使用 ifconfig 时仍没有看到该接口起动，而是看到了以下错误消息？

```
SIOCGIFFLAGS: No such device
```

答：此错误表示相关模块没有正确加载。请尝试手动插入相关模块并查看在模块加载期间是否有错误。

问：如何确定拥有的 WiLink 驱动程序和固件版本？答：若要查找 WiLink™ 固件（通常被称为 wl18xx-fw-x.bin）的版本，请在启动并以 root 身份登录后，在 Sitara 串行终端中输入以下命令：

```
grep Rev /lib/firmware/ti-connectivity/wl18xx-fw-4.bin
```

重要声明和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、某特定用途方面的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他功能安全、信息安全、监管或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的应用。严禁对这些资源进行其他复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。您应全额赔偿因在这些资源的使用中对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，TI 对此概不负责。

TI 提供的产品受 TI 的销售条款或 ti.com 上其他适用条款/TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

邮寄地址：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265