Tina Linux Wi-Fi 模组
移植指南

版本号: 1.9
发布日期: 2024.12.06

版本历史

版本号	日期	制/修订人	内容描述
1.0	2022.05.20	AWA1381	初始版本
1.1	2023.04.20	AWA1381	1. 第二章，兼容 tina5.0 部分说明。
1.2	2023.09.26	AWA1888	1. 第二章，增加 USB 模组说明。2. 第三章，增加 AIC8800D USB 移植示例。
1.3	2023.10.23	AWA1381	1. 修改部分错误标点使用。
1.4	2023.11.04	AWA1436	1. 增加 AIC8800D40 移植示例。
1.5	2023.12.12	AWA1436	1. 增加适用 buildroot 编译方式相关说明。
1.6	2024.1.2	AWA1888	1. 增加 aic8800 buildrootd 移植示例。
1.7	2024.4.28	AWA1888	1. 修改硬件平台适用范围说明。
1.8	2024.4.28	KPA0568	1. 添加 tina5.0 以及新内核支持。
1.9	2024.12.06	AWA2255	1. 新增 AIC8800D40/D80 USB 模组移植说明。2. 修改 Tina5.0 相关文件路径说明。3. 修改设备树示例。4. 修改硬件平台适用范围说明。5. 修改 XRADIO 系列模组介绍。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

目　录

版权所有 © 珠海全志科技股份有限公司。保留一切权利

版权所有 © 珠海全志科技股份有限公司。保留一切权利

ALLWINNER®

版权所有 © 珠海全志科技股份有限公司。保留一切权利

插　图

版权所有 © 珠海全志科技股份有限公司。保留一切权利

版权所有 © 珠海全志科技股份有限公司。保留一切权利

版权所有 © 珠海全志科技股份有限公司。保留一切权利

表　格

版权所有 © 珠海全志科技股份有限公司。保留一切权利

1 前言

1.1 文档简介

本文档主要介绍 Tina Linux 平台上 Wi-Fi 模组移植的基本原理和验证方法，以指引开发者能快速的在 Tina Linux 平台上适配一款新的 Wi-Fi 模组。

文档结构如下：

表 1-1: 文档结构

章节	说明
第一章：前言	适用范围和相关术语说明，简要浏览。
第二章：移植说明	Wi-Fi 框架，原理，移植步骤说明，需要重点关注。
第三章：示例说明	具体模组移植示例，供参考。
第四章：验证说明	移植完成，功能验证说明，简要浏览。
第五章：排查说明	移植难点分析，排查问题环节，需要重点关注。

1.2 目标读者

- Tina Linux Wi-Fi 模块开发工程师。
- Tina Linux Wi-Fi 模块测试工程师。

1.3 适用范围

主控：R/V/MR/T/D1/H 等系列。

模组：XR819、XR819S、XR829、AW859、AW869、RTL8723DS、AP6212、ESP32、AIC8800。

系统：Tina3.5 及以上。

内核：linux-4.9，linux-5.4，linux-5.15，linux-5.10，linux-6.6。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

1.4 文档约定

1.4.1 标志说明

⚠ 注意

- **提醒操作中应注意的事项。不当的操作可能会损坏器件，影响可靠性、降低性能等。**

说明

为准确理解文中指令、正确实施操作而提供的补充或强调信息。

技巧

一些容易忽视的小功能、技巧。了解这些功能或技巧能帮助解决特定问题或者节省操作时间。

1.4.2 地址与数据描述方法约定

本文档在描述地址、数据时遵循如下约定：

表 1-2: 地址与数据描述方法约定

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

1.4.3 数值单位约定

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

表 1-3: 数值单位约定

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

版权所有 © 珠海全志科技股份有限公司。保留一切权利

类型	符号	对应数值
	1 M	1 000 000
	1 G	1 000 000 000

1.5 相关术语介绍

1.5.1 软件术语

表 1-4: 软件术语

术语	解释说明
802.11	无线通信协议。
station	Wi-Fi 工作 sta 模式，类比手机主动连网。
ap	Wi-Fi 工作 ap 模式，类比手机开热点。
monitor	Wi-Fi 工作 monitor 模式，类比抓包网卡。
wifimanager	Wi-Fi 最上层应用，管理 sta 模式。
softap	Wi-Fi 最上层应用，管理 ap 模式。
smartlink	Wi-Fi 最上层应用，管理配网模式。
SDIO/USB	Wi-Fi 和主控之间通信的一种接口与协议。
firmware	运行在 Wi-Fi 芯片上的代码，各种 bin 文件【原厂提供, 不开源】。
rf 指标	射频参数，如发射功率、频偏等。
sys_config.fex/dts	硬件资源配置文件，包括供电，gpio 的配置。

说明

最新的 wifimanager2.0 将 sta,ap,monitor, 配网集成到一起。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

2 移植说明

本章总体先从 Wi-Fi 系统软件架构、移植原理、移植步骤等方面进行阐述说明，使读者对整个 Wi-Fi 体系结构以及移植原理有一个初步认识。如无特殊说明默认是指在 tina 的 opewrt 编译系统进行的移植。

2.1 框架概述

Wi-Fi 是一种无线通信技术，在 Tina Linux 系统上一般可处于三种工作模式，分别是: STATION、AP、MONITOR。

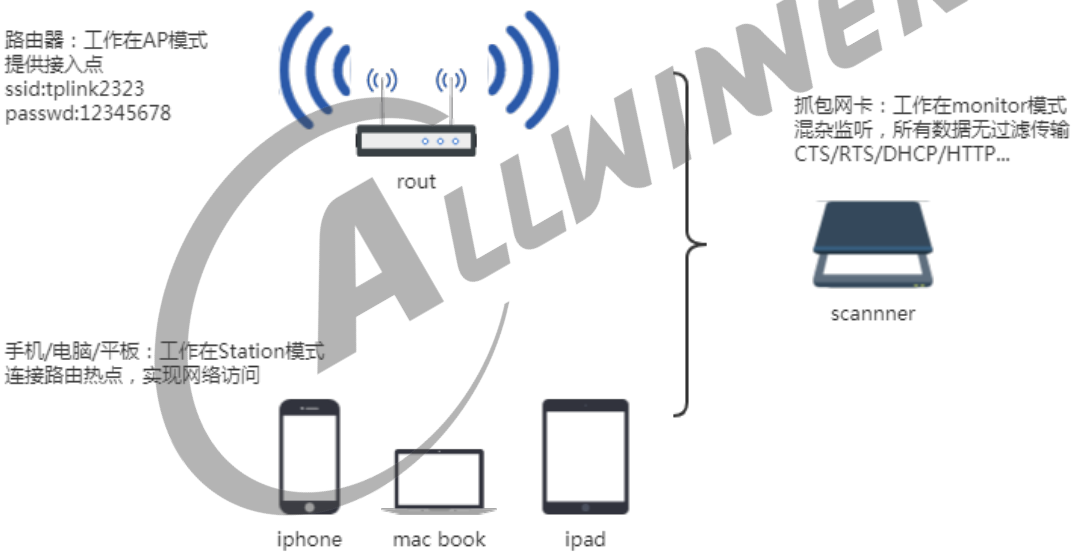


图 2-1: wifi 工作模式

遵循 802.11 协议和 TCP/IP 网络分层模型：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

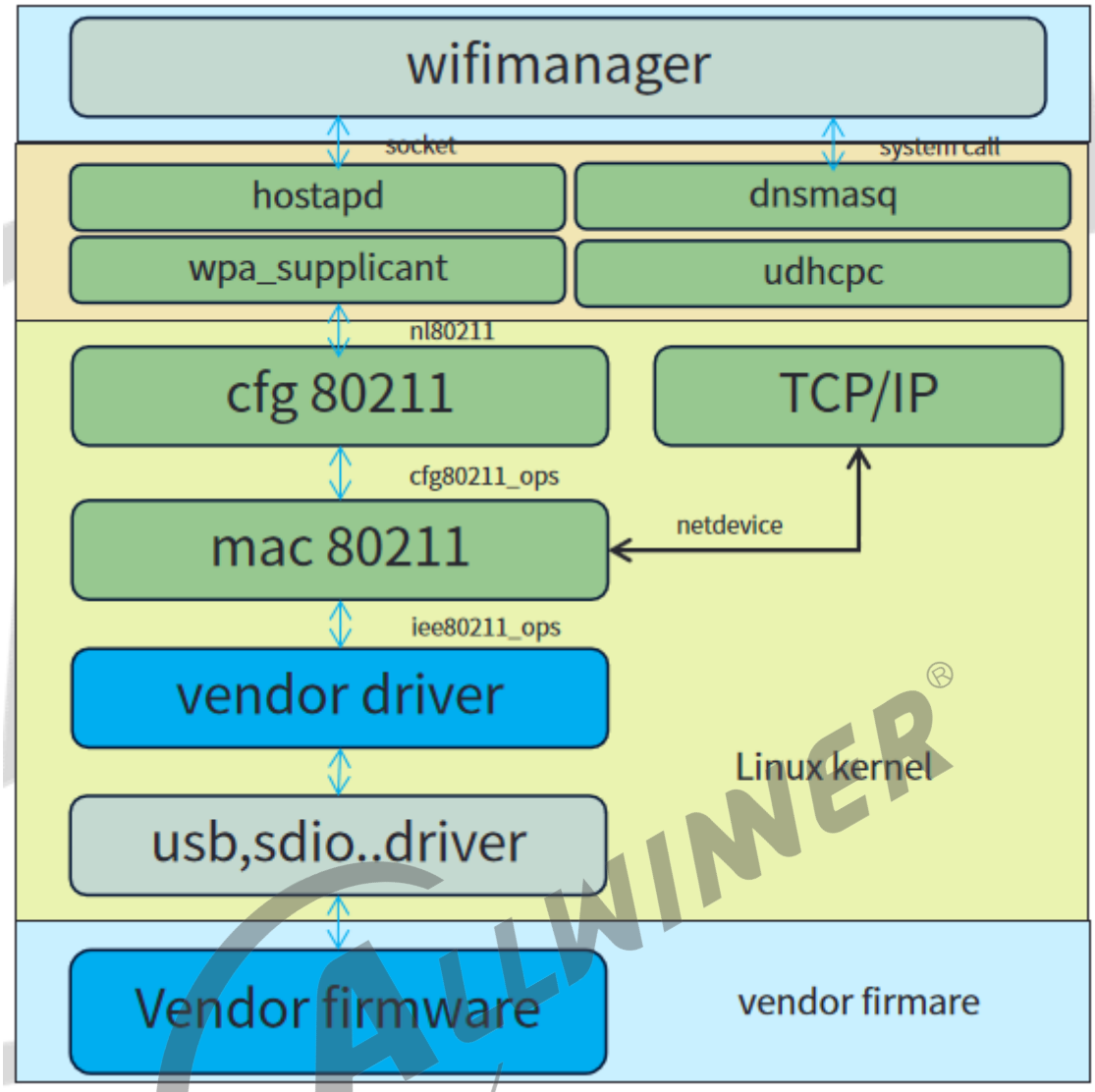


图 2-2: wifi 分层模型

说明

以上分层模型可能会存在重叠部分，详细的分层可以参考 TCP/IP 协议模型。

- wifimanger：主要用于 STATION 模式，提供 wifi 连接扫描等功能。
- softap manager：提供启动 AP 的功能。
- smartlink：对于 NoInput 的设备，通过借助第三方设备（如手机）实现透传配网的功能。
- wpa_supplicant: 开源的无线网络配置工具，主要用来支持 WEP，WPA/WPA2 和 WAPI 无线协议和加密认证的，实际上的工作内容是通过 socket 与驱动交互上报数据给用户。
- hostapd: 是一个用户态用于 AP 和认证服务器的守护进程。
- monitor: wifi 处于混杂设备监听模式的处理应用。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

2.2 原理概述

从整体结构上 Wi-Fi 可以分为设备端（Device）和主机端（Host）；设备端主要是数据链路层的实现和接口管理，主机端主要是应用层和协议层的实现。

模组驱动移植主要所做的工作：供电使能，传输接口适配。

- 供电使能：提供 WiFi 供电，使能操作。如果平台 WiFi 供电通过电源管理芯片，那么就需要配置电源管理芯片电源域进行使能 WiFi 的所需电；WiFi 芯片正常工作需要主控 soc 输出使能信号。
- 传输接口适配：WiFi 通过通信接口与主控进行连接，常见的通信接口有 SDIO,USB,PCI,SPI 等。通信接口的驱动往往是由主控端提供，平台层需要提供通信接口初始化（如 SDIO 扫卡）、读、写等函数操作集合。一般情况下由于通信接口都是标准的 Linux API，所以往往模组原厂已经适配好了读写等操作的接口函数，平台适配可能仅仅需要配置平台差异就好，如选择使用 SDIO 接口，需要确定选择的 SDIO 卡号、扫卡函数等。

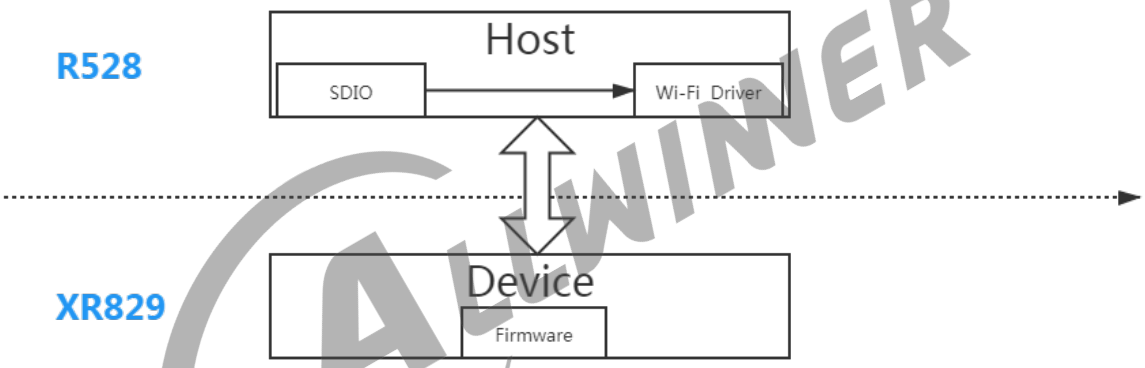


图 2-3: wifi 设备模型

本章节以围绕硬件接线图及 WiFi 工作的硬件条件、WiFi 启动流程进行重点说明，以理清移植模组本质和原理。

2.2.1 硬件连接简图

这里主要介绍的是 SDIO、USB 通信方式的连接图：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

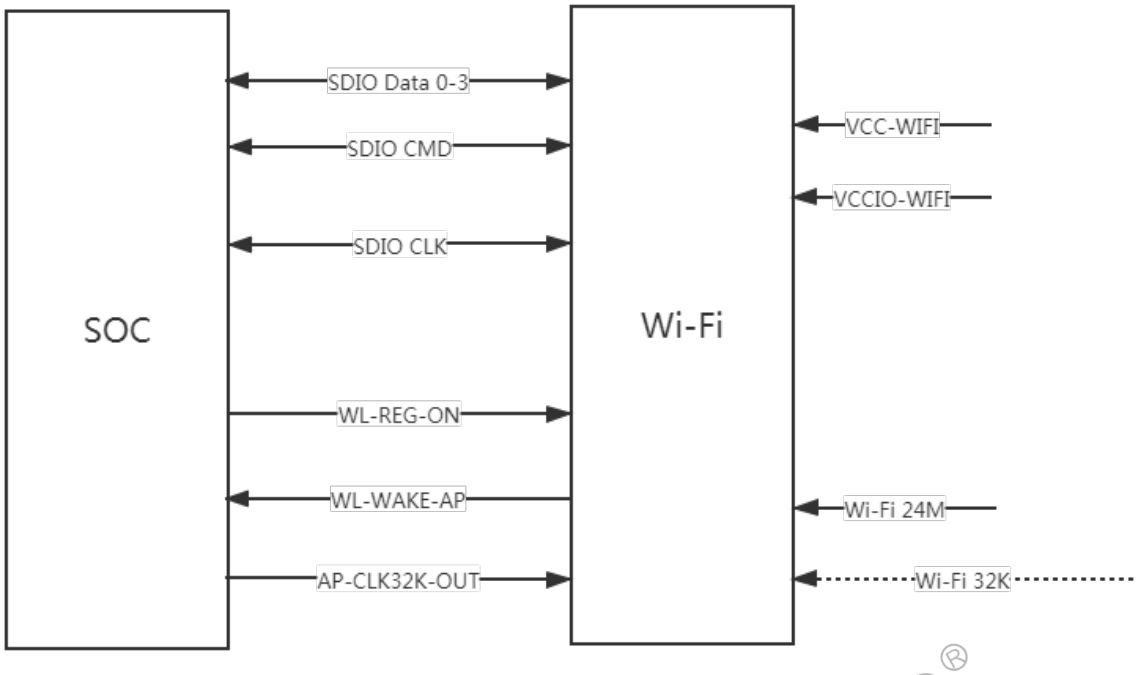


图 2-4: 主控和 wifi 硬件连接简图

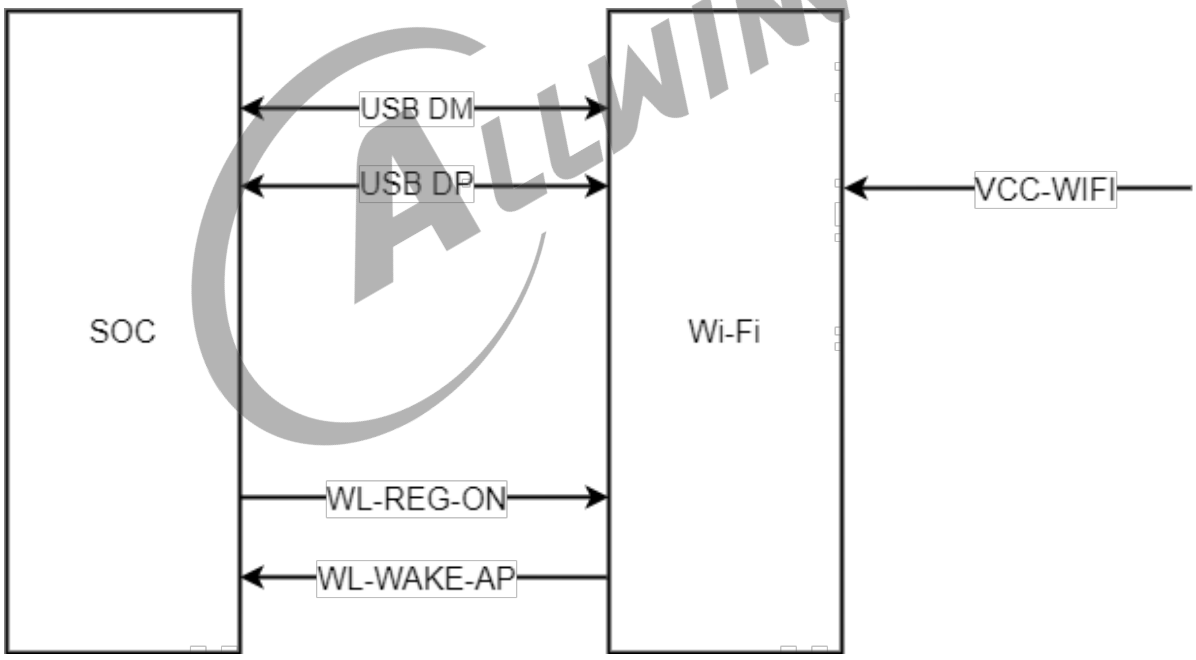


图 2-5: 主控和 wifiUSB 硬件连接简图

2.2.2 硬件工作条件

SDIO Wi-Fi 模组工作的条件，如下图，需要满足以下几个条件:

（1）供电：一般有两路供电，其中 VCCWIFI 为主电源，VCCIOWIFI 为 IO 上拉电源。【需要特别

版权所有 © 珠海全志科技股份有限公司。保留一切权利

留意，有时候设计可以两路电合并】

（2）使能：要能正常工作，需要 WLREGON 给高电平。【需要特别留意，部分模组可能有时序要求，比如先拉高再拉低，再拉高】

（3）唤醒主控：当系统休眠时，Wi-Fi 模组可通过 WLWAKEAP 的中断唤醒主控。【需要特别留意，有些模组也通过该引脚来作为主控接收数据的中断】

（4）SDIO：与 SOC 的通信有通过 USB，SDIO 等，这里以 SDIO 为例，其中 SDIO 0~3 为 SDIO 的 4 条数据线。

（5）次时钟 32.768Khz 信号：根据模组而定，有些模组内部通过（6）中的输入的 clk 进行分频得到，有些需要外部单独输入该信号。

（6）主时钟 24/26Mhz 信号。

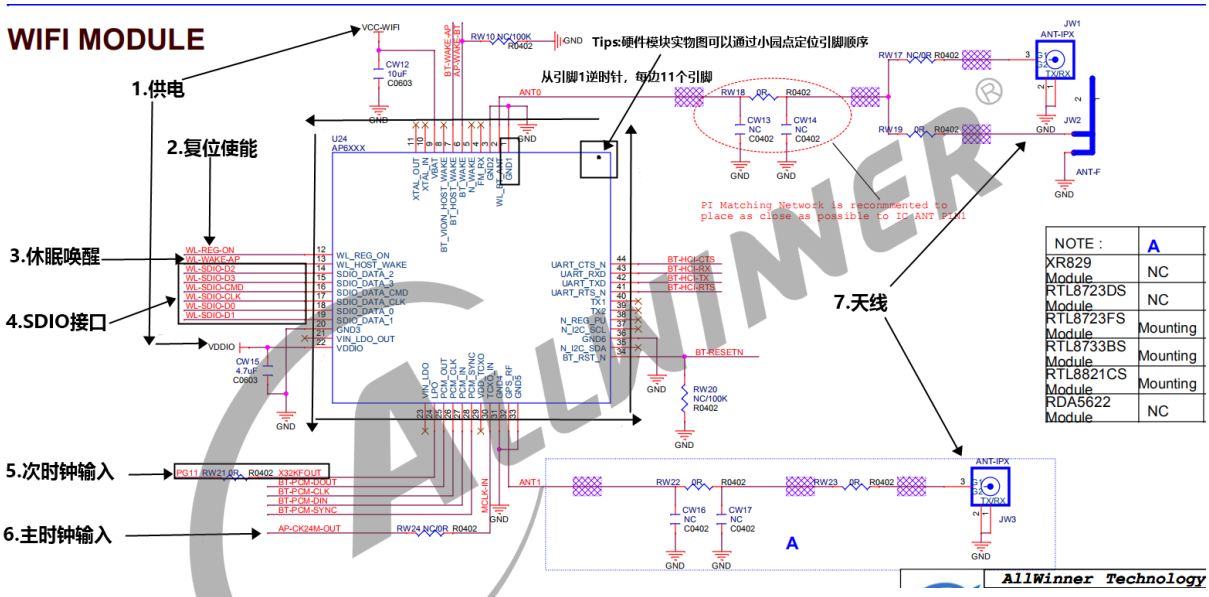


图 2-6: 模组引脚原理图

USB Wi-Fi 模组工作的条件，如下图，需要满足以下几个条件:

（1）供电：一般只有一路电源，采用贴板的 USB 模组一般会使用 VCC-WiFi 给模组供电，电源由软件配置开启。采用 USB 接口可拔插设计的 WiFi 模组一般使用 USB 供电，一般为常电，则无需使用软件配置开启电源。

（2）使能：要能正常工作，需要 WLREGON 给高电平。采用 USB 接口可拔插设计的 WiFi 模组可能无此引脚则无需配置。【需要特别留意，部分模组可能有时序要求，比如先拉高再拉低，再拉高】

（3）唤醒主控：当系统休眠时，Wi-Fi 模组可通过 WLWAKEAP 的中断唤醒主控。USB 模组一般使用 USB 唤醒无须配置，若模组特殊要求使用 GPIO 唤醒则需要配置。【需要特别留意，多数 USB WiFi 模组不支持休眠唤醒】

版权所有 © 珠海全志科技股份有限公司。保留一切权利

（4）USB：与 SOC 的通信有通过 USB，SDIO 等，这里以 USB 为例，其中 DP、DM 为 USB 的两条数据线。

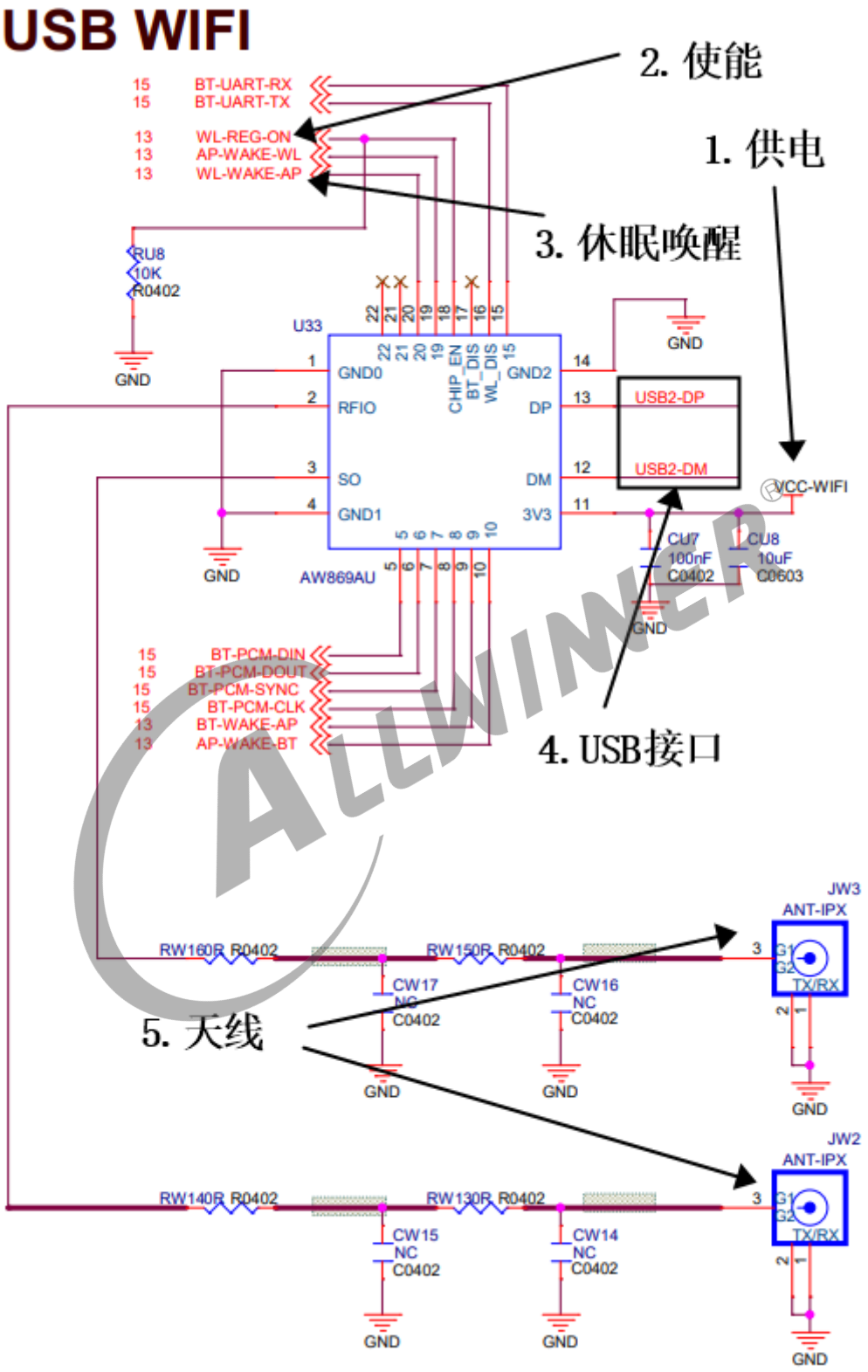


图 2-7: USB 模组引脚原理图

版权所有 © 珠海全志科技股份有限公司。保留一切权利

2.2.3 软件启动流程

整体流程分为如下几个阶段：系统启动-> 驱动加载-> 服务加载-> 启动网卡-> 访问网络。

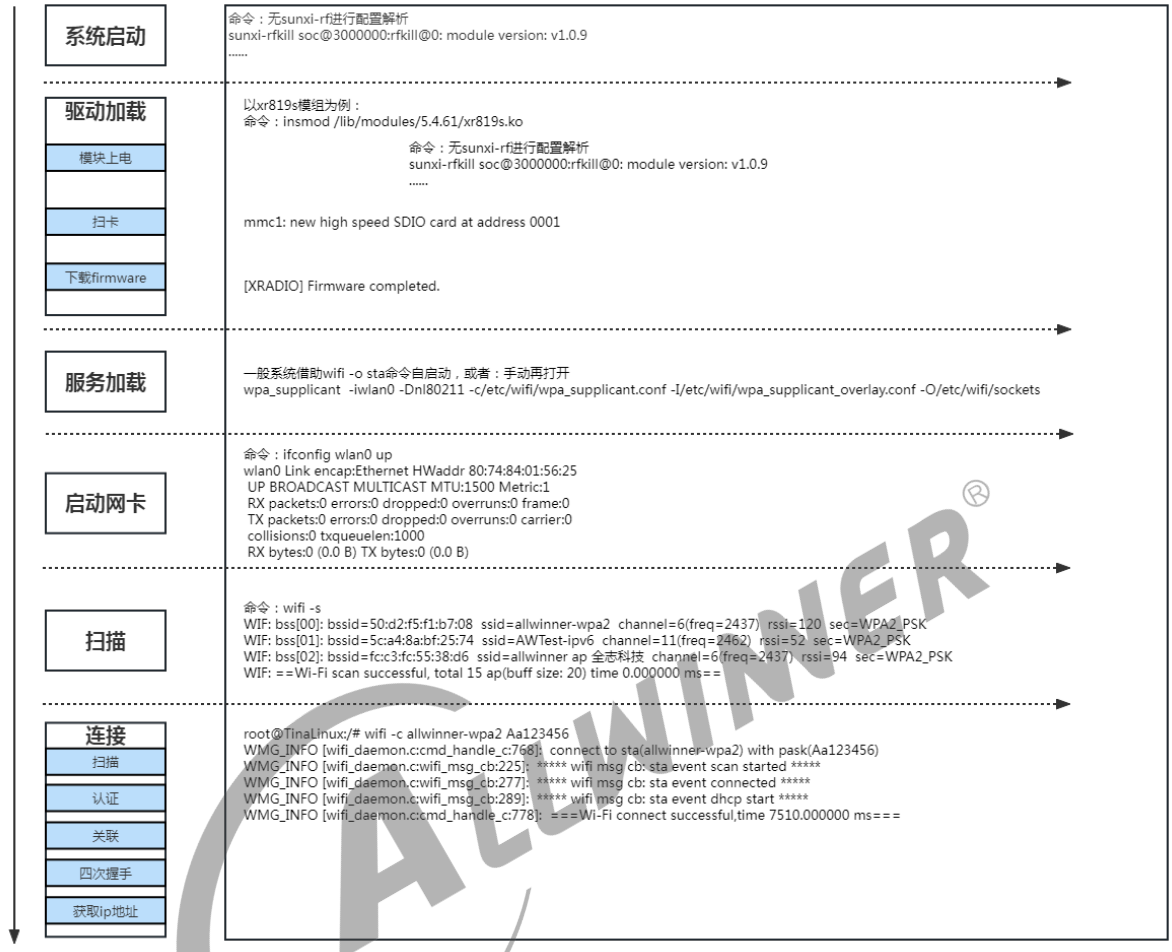


图 2-8: Wi-Fi 启动流程

2.3 步骤概述

移植适配 Wi-Fi 模组，本小节详细描述移植一款新物料的步骤。总结为五步法，接下来将对其 5 步依次展开叙述。

- 内核驱动适配
- 硬件资源适配
- 方案 module 适配
- 添加 Firmware
- 应用工具适配

版权所有 © 珠海全志科技股份有限公司。保留一切权利

表 2-1: Tina4.0 SDK 相关文件路径

文件	路径	说明
Wi-Fi 驱动	Tina4.0/lichee/linux-x.x/drivers/net/wireless	驱动源码
firmware	Tina4.0/package/firmware/linux-firmware	firmware 文件
board.dts	Tina4.0/device/…/linux/board.dts	硬件资源配置
modules.mk	Tina4.0/target/allwinner/xxx-common/modules.mk	方案配置
wpa_supplicant	Tina4.0/package/network/services/hostapd	网络服务
rf 测试工具	Tina4.0/package/utils/rftest	射频测试工具

表 2-2: Tina5.0 SDK 相关文件路径

文件	路径	说明
Wi-Fi 驱动	Tina5.0/bsp/drivers/net/wireless	驱动源码
fw mk	Tina5.0/openwrt/openwrt/package/firmware/linux-firmware/xx.mk	fw.mk 文件
fw bin	Tina5.0/platform/allwinner/wireless/firmware	fw 源文件
board.dts	Tina5.0/device/config/chips/xxx/configs/xxx/board.dts	硬件资源配置
modules.mk	Tina5.0/openwrt/target/xxx/xxx-common/modules.mk	方案 module 配置
wpa_supplicant	Tina5.0/openwrt/openwrt/package/network/services/hostapd	网络服务
rf 测试工具	Tina5.0/openwrt/package/feeds/utils/rftest	射频测试工具

说明

Tina5.0 最大差异点，将每个包独立一个仓库，同时把源码和 Makefile 编译文件拆开。
Tina5.0 中 buildroot 编译方式暂未支持 rf 测试工具。

2.3.1 内核驱动适配

内核驱动适配又分为 5 个小步骤：

1. 获取驱动源码

在新方案上移植一款 Wi-Fi 模组时，可以先查看《支持列表》文档是否有类似方案支持该模组。如果没有支持，则需要跟模组原厂获取驱动资料包。拿到原厂提供的资料包需要仔细阅读模组原厂提供的材料，如：支持的通信接口类型 SDIO, 还是 USB；SDIO 是 2.0 还是 3.0；支持 2.4G 还是 5G 甚至是双频等。

2. 添加 Kconfig 和 Makefile 配置。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

顶层 Kconfig 文件中引入 xxx 驱动的 Kconfig 配置。

```
Tina4.0/lichee/linux-5.4/drivers/net/wireless/Kconfig
Tina5.0/bsp/drivers/net/wireless/Kconfig

source "drivers/net/wireless/xxx/Kconfig"
```

顶层 Makefile 文件中引入 xxx 驱动的 Makefile 配置。

```
Tina4.0/lichee/linux-5.4/drivers/net/wireless/Makefile
Tina5.0/bsp/drivers/net/wireless/Makefile

obj-$(CONFIG_RTL8723DS)  += xxx/
```

说明

xxx 代办具体驱动型号，如 XR829、rtl8723ds 等。

3. 修改驱动源码

这一步是很关键的，我们需要定义自己的模组上下电，扫卡接口来对接 Tina 系统提供的接口，以及原厂提到的关于平台适配的注意事项修改。

Tina 常用接口：

```
void sunxi_wlan_set_power(bool on);      //上下电接口，参数0:上电；1：下电
int  sunxi_wlan_get_bus_index(void);     //获取sdio通信接口的busnum，一般是sdc1，USB模组无须配置
int  sunxi_wlan_get_oob_irq(void);       //获取中断号
int  sunxi_wlan_get_oob_irq_flags(void);  //获取中断标志
void sunxi_mmc_rescan_card(unsigned ids);  //mmc提供的扫卡函数,USB模组无须扫卡
```

上电：

```
int platform_wifi_power_on(void)
{
    int wlan_bus_index = 0;
    sunxi_wlan_set_power(1);
    mdelay(100);

    wlan_bus_index = sunxi_wlan_get_bus_index();
    if(wlan_bus_index < 0){
        RTW_INFO("get wifi_sdc_id failed\n");
        return -1;
    } else {
        RTW_INFO("----- %s sdc_id: %d\n", __FUNCTION__, wlan_bus_index);
        sunxi_mmc_rescan_card(wlan_bus_index);
    }
#ifdef CONFIG_GPIO_WAKEUP
    oob_irq = sunxi_wlan_get_oob_irq();
#endif
    return 0;
}
```

下电：

```
void platform_wifi_power_off(void)
{
    int wlan_bus_index = 0;
    sunxi_wlan_set_power(0);
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    mdelay(100);
    RTW_INFO("%s: remove card, power off.\n", __FUNCTION__);
    wlan_bus_index = sunxi_wlan_get_bus_index();
    sunxi_mmc_rescan_card(wlan_bus_index);
}
```

提供 platform_wifi_power_on() 和 platform_wifi_power_off() 接口给驱动 probe 调用。

说明

以上提到的接口对接都是系统平台通用的，可以直接从已经支持的模组中获取适配的文件。drivers/net/wireless/xxx/platform/platform_ARM_SUNxI_sdio.c

2.3.2 硬件资源适配

1. 确认供电方式

模组的供电，我们需要确认的是 VCC-WIFI 和 VCCIO-WIFI 供电的来源，如果两路供电来源来自电源管理芯片，则需要配置电源管理芯片输出两路电，如果是直接供电，那么就不需要进行软件配置输出。

两路供电来源于 R818 AXP707：

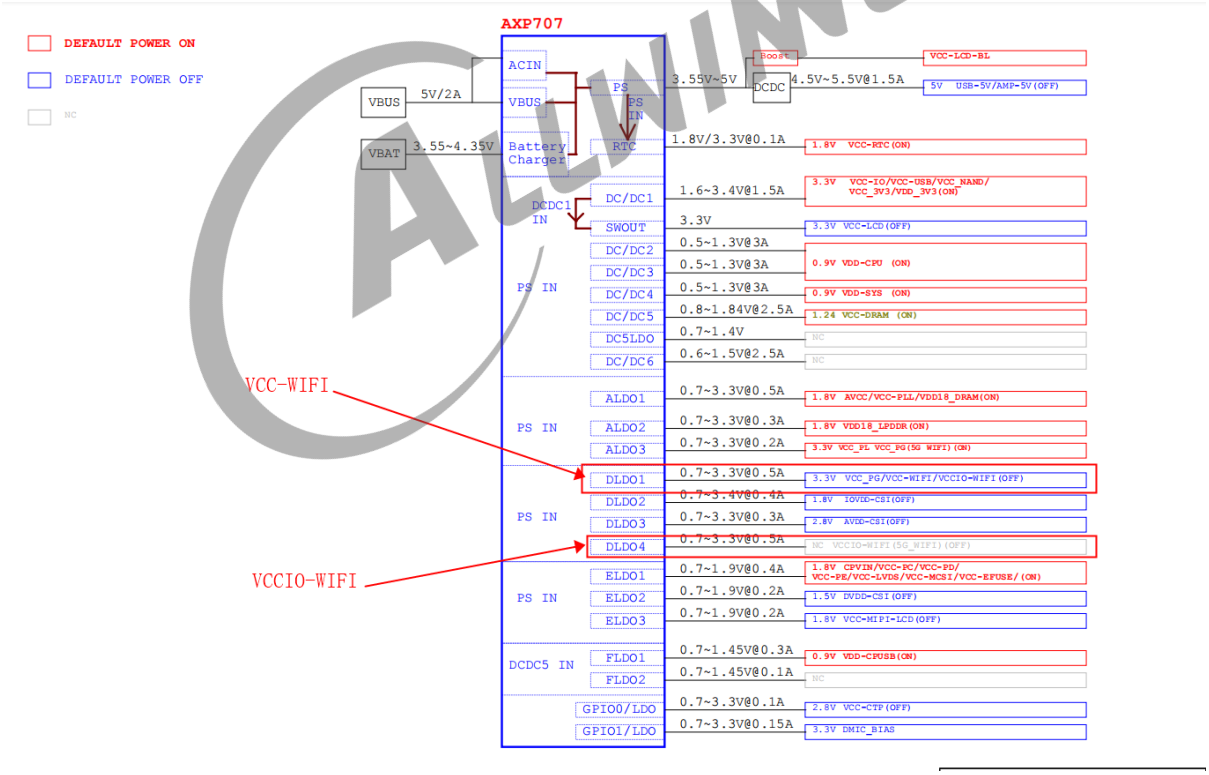


图 2-9: wifi 两路供电

合并一路供电来源于 R329：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

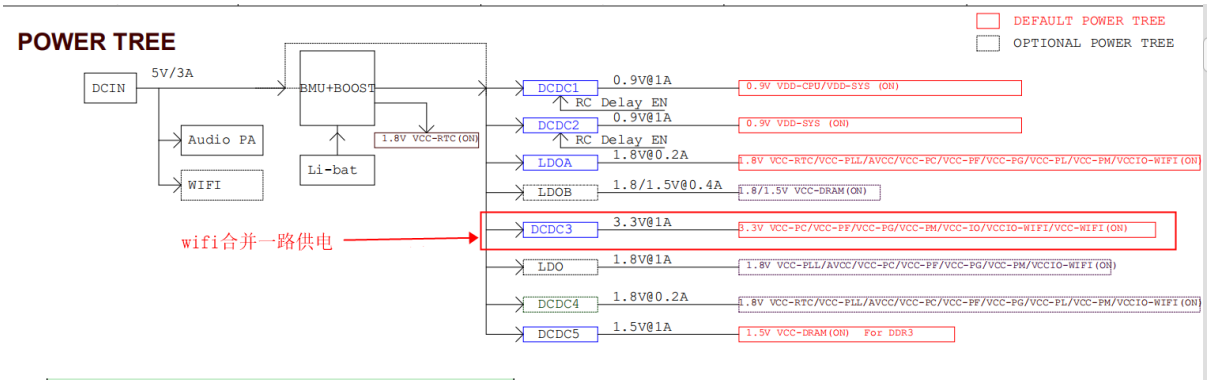


图 2-10: wifi 合并一路供电

对于 WiFi 和 BT 二合一的模组，供电部分是复用的。分为为 VCC-WIFI 和 VCCIO-WIFI，从 R818 原理图来判断，两路电源是由电源管理芯片 DLDO1 和 DLDO4 分别输出；从 R329 原理图来判断，合并一路供电由 DCDC3 直接供电。关于电源管理芯片的驱动原理可参考电源管理驱动使用说明或咨询电源管理芯片驱动开发工程师。通常情况下该部分已经准备好，对于 Wi-Fi 开发工程师来说只需要会配置电源输出即可, 配置方法见下 board.dts 中关于 wlan_power、wlan_io_regulator 标识的配置。

2.3.3 方案 module 适配

1. 添加 module 配置文件：

模组一般会编译生成 KO 文件，添加这个配置文件主要是基于 Tina 系统的构建法则，将编译生成的 ko 文件拷贝到文件系统中，同时可以配置驱动是否需要开机启动自加载。示例如下：

```
define KernelPackage/net-xr829-40M
  SUBMENU:=$(WIRELESS_MENU)
  TITLE:=xr829 support (staging)
  DEPENDS:= +@IPV6
  KCONFIG:= \
        CONFIG_XR829_WLAN=m \
  FILES:=$(LINUX_DIR)/drivers/net/wireless/xr829/wlan/xradio_core.ko
  FILES+=$(LINUX_DIR)/drivers/net/wireless/xr829/wlan/xradio_wlan.ko
  FILES+=$(LINUX_DIR)/drivers/net/wireless/xr829/umac/xradio_mac.ko
  AUTOLOAD:=$(call AutoProbe, xradio_mac xradio_core xradio_wlan)
endef

define KernelPackage/net-xr829-40M/description
 Kernel modules for xr829 support
endef

$(eval $(call KernelPackage,net-xr829-40M))
```

module 配置整体分为两个部分，都是固定的格式，第一部分是配置 module 在 menuconfig 中的显示和加载信息，第二部分是 module 的简单描述信息。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

表 2-3: module 配置说明

标签	说明
SUBMENU	用于配置该 module 显示在 menuconfig 的那个菜单下。
TITLE	用于配置该 module 在 menuconfig 的提示信息。
DEPENDS	用于添加 Tina 配置的依赖选项。【类似 select 的作用】
KCONFIG	用于添加内核配置的依赖选项。【类似 select 的作用】
FILES	添加具体内核驱动 ko 的路径。【可以有多个 ko】
AUTOLOAD	配置需要自加载的 ko。【可以有多个 ko】

说明

上述方式只有 openwrt 编译方式需要添加，buildroot 编译方式不需要添加。
buildroot 方式编译的 ko 文件会统一拷贝到 rootfs 文件系统中，无法配置文件，但 buildroot 方式不支持 wifi 内核模块自加载，需另写脚本自加载，或系统起来后手动加载。

2.3.4 添加 Firmware

1. 添加 firmware 文件：

部分 Wi-Fi 模组工作涉及运行在 devices 端的 firmware 文件【可以简单理解为一些未开源的 bin 文件】，添加这个文件主要就是在启动阶段中完成下载的流程，具体 firmware 的原理不用深入了解。

Tina 上的统一路径为：

```
Tina4.0/package/firmware/linux-firmware
Tina5.0/openwrt/openwrt/package/firmware/linux-firmware/xx.mk
Tina5.0/platform/allwinner/wireless/firmware/xxx源码
```

2. 添加 firmware 配置：

增加 Makefile 文件，配置的主要作用就是基于 Tina 系统的构建法则将 firmware 文件拷贝到系统对应的目录，为启动流程下载 firmware 做准备。

2.3.5 应用工具适配

1.iperf

2.wifimanager

3.wpa_supplicant

4.wpa_cli

5.rf 工具（buildroot 编译方式暂未支持）

版权所有 © 珠海全志科技股份有限公司。保留一切权利

应用工具的配置，主要功能就是借助应用来测试 Wi-Fi 的扫描，联网等功能。

说明

示例说明章节都是基于 Tina4.0 开发举例的，关于 Tina5.0 的相关示例，只要清楚对应代码的路径，修改方法完全一致。另外关于 wifimanager2.0 的配置使用请参考《Tina_Linux_Wi-Fi 软件开发指南》。

2.4 模组概述

表 2-4: 支持列表

Vendor	PartNumber	Interface	Wi-Fi	BT
Allwinner	XR819	SDIO	b/g/n	不支持
Allwinner	XR829	SDIO	b/g/n	支持
Allwinner	XR819S	SDIO+UART	b/g/n	支持
Allwinner	AW869A	SDIO+UART	a/b/g/n/ac	支持
Realtek	rtl8723bs	SDIO+UART	b/g/n	支持
Realtek	rtl8723ds	SDIO+UART	b/g/n	支持
Realtek	rtl8723cs	SDIO+UART	b/g/n	支持
Realtek	rtl8733bs	SDIO+UART	b/g/n	支持
Realtek	rtl8821cs	SDIO+UART	b/g/n/ac	支持
Realtek	rtl8822cs	SDIO+UART	b/g/n/ac	支持
Realtek	rtl8189fs	SDIO+UART	b/g/n	支持
Realtek	rtl8188fu	SDIO+UART	b/g/n	支持
ESPRESSIF	esp32	SDIO	b/g/n	支持

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3 示例说明

3.1 XRADIO 系列

目前 XRADIO 有 XR819/XR829/XR819S 三款自研的无线模组：

表 3-1: xradio 支持列表

Vendor	PartNumber	Interface	Wi-Fi	BT
Allwinner	XR819	SDIO	b/g/n	不支持
Allwinner	XR829	SDIO	b/g/n	支持
Allwinner	XR819S	SDIO+UART	b/g/n	支持

其驱动的代码结构基本保持一致，所以我们将以 XR829 为例子详细介绍模组适配的全过程。

3.1.1 XR829 模组移植

主控：R528

无线模组：XR829

系统版本：Tina4.0 linux-5.4

方案：r528_evb1-tina

3.1.1.1 内核驱动适配

内核驱动适配分为如下四个步骤：

- 获取源码。
- 添加 Kconfig 和 Makefile 配置。
- make kernel_menuconfig 配置。
- 编译。

1）获取 XR829 驱动源码，放到内核驱动路径下。客户可以从一号通账号获取。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
Tina4.0/lichee/linux-5.4/drivers/net/wireless/xr829/
├──── include
├──── Kconfig
├──── Makefile
├──── umac
└──── wlan
```

2）内核添加 Kconfig 和 Makefile 的配置。

Tina4.0/lichee/linux-5.4/drivers/net/wireless/Kconfig 文件中引入 XR829 驱动的 Kconfig 配置。

```
source "drivers/net/wireless/xr829/Kconfig"
```

Tina4.0/lichee/linux-5.4/drivers/net/wireless/Makefile 文件中引入 XR829 驱动的 Makefile 配置。

```
obj-$(CONFIG_XR829_WLAN)   += xr829/
```

3）make kernel_menuconfig 配置

在 Tina4.0/lichee/linux-5.4 内核目录执行 m kernel_menuconfig。

a. XR829 驱动的配置

```
> Device Drivers > Network device support > Wireless LAN
    <M>   XR829 WLAN support
```

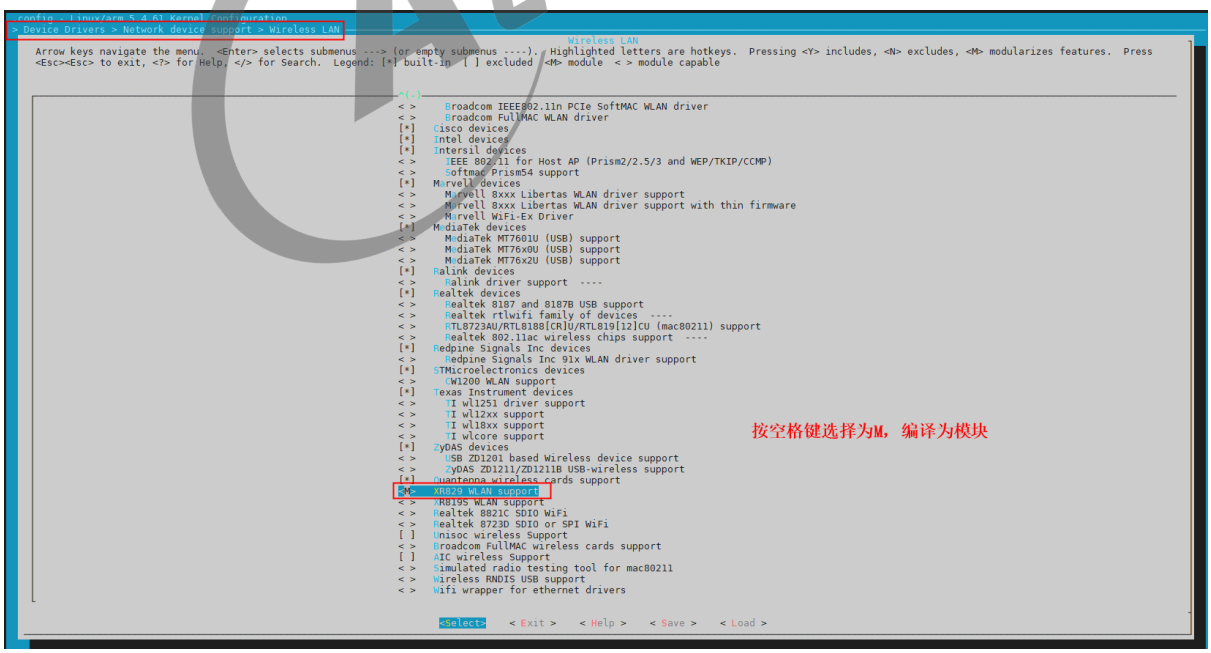


图 3-1: XR829 内核配置

b. sunxi-rf 的配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

> Device Drivers > Misc devices
  <*> Allwinner rfkill driver

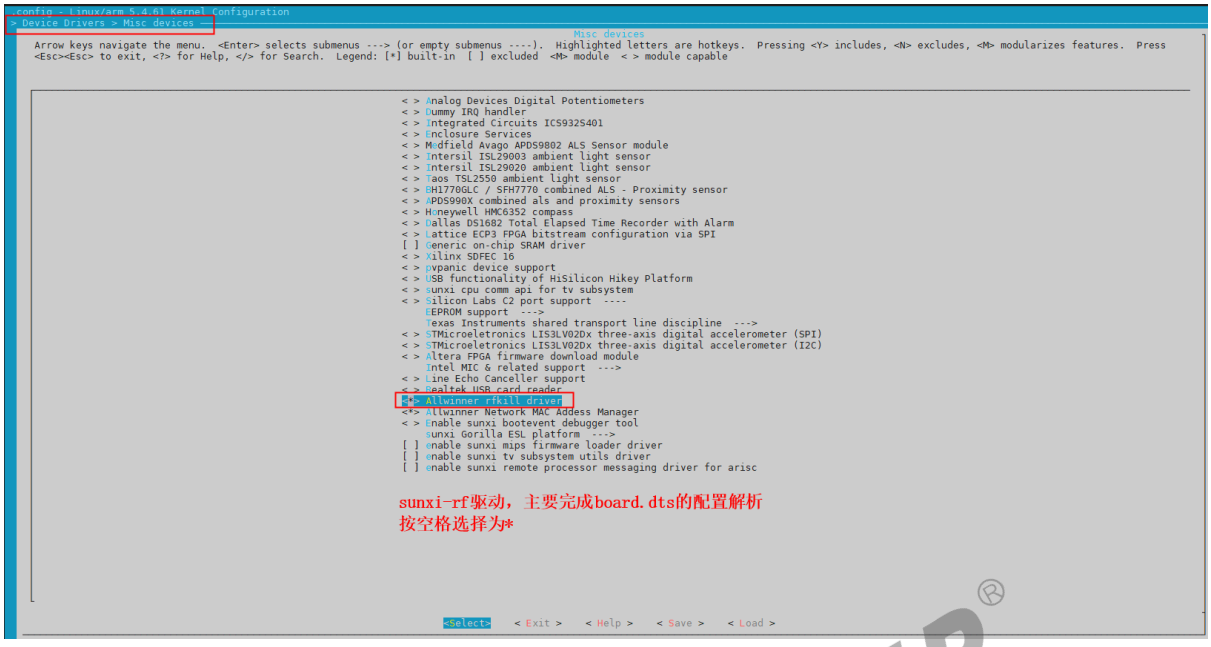


图 3-2: Tina-sunxi-rf 配置

c. SDIO 的配置

> Device Drivers > MMC/SD/SDIO card support
  <*> Allwinner sunxi SD/MMC Host Controller support --->

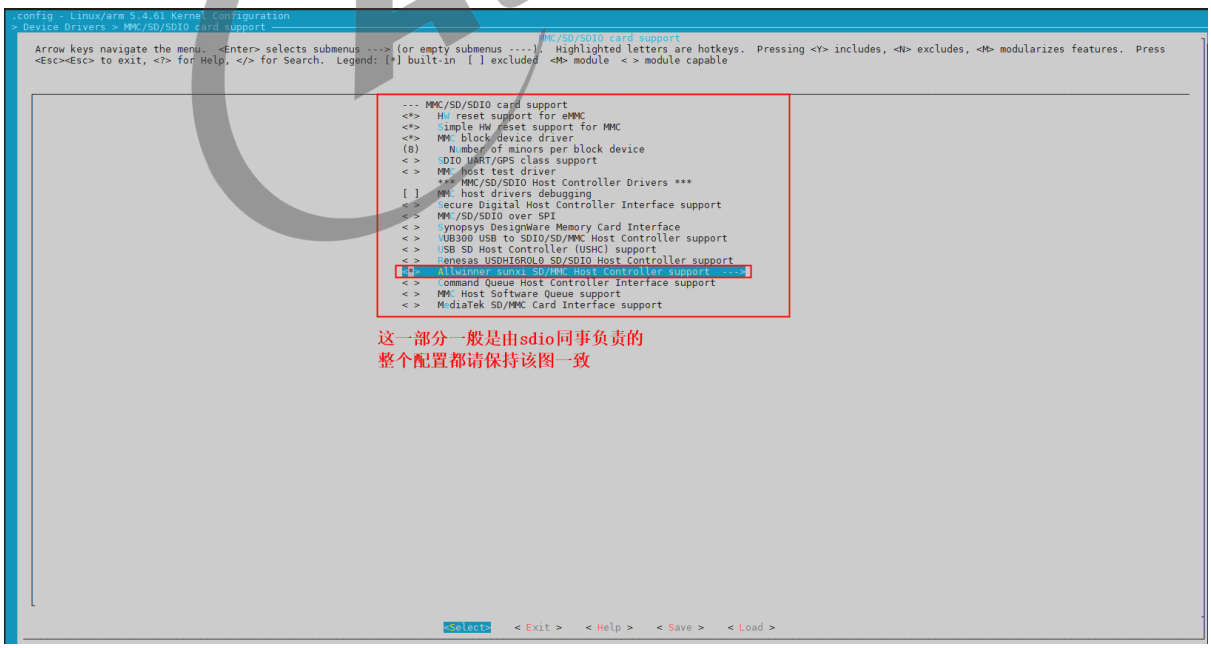


图 3-3: Tina-sdc 配置

4) 编译

版权所有 © 珠海全志科技股份有限公司。保留一切权利

在 Tina4.0/lichee/linux-5.4 内核目录执行 mkernel 进行编译。

Tina4.0/lichee/linux-5.4]$ mkernel

```
Tina4.0/lichee/linux-5.4/drivers/net/wireless/xr829/
├──── include
├──── Kconfig
├──── Makefile
├──── modules.order
├──── umac
├──── wlan
├──── xr829.ko        //正常编译出xr829.ko模块
├──── xr829.mod
├──── xr829.mod.c
├──── xr829.mod.o
└──── xr829.o
```

3.1.1.2 硬件资源适配

在 Tina4.0/device/config/chips/r528/configs/evb1/linux/board.dts 中添加引脚配置。

```
&pio {
    ...
    wlan_pins_a:wlan@0 {
            pins = "PG11";
            function = "clk_fanout1";  //这里涉及一个特殊处理，该方案的32k采用PG11复用给出
        };
};

&soc {
    ...
    rfkill: rfkill@0 {
        compatible   = "allwinner,sunxi-rfkill";
        chip_en;
        power_en;
        pinctrl-0 = <&wlan_pins_a>;
        pinctrl-names = "default";
        status      = "okay";

        wlan: wlan@0 {
            compatible   = "allwinner,sunxi-wlan";
            clock-names = "32k-fanout1";
            clocks = <&ccu CLK_FANOUT1_OUT>;
            wlan_busnum   = <0x1>;
            wlan_regon   = <&pio PE 17 GPIO_ACTIVE_HIGH>;
            wlan_hostwake  = <&pio PG 10 GPIO_ACTIVE_HIGH>;
            /*wlan_power   = "VCC-3V3";*/
            /*wlan_power_vol = <3300000>;*/
            /*interrupt-parent = <&pio>;
            interrupts = < PG 10 IRQ_TYPE_LEVEL_HIGH>;*/
            wakeup-source;

        };
    };
};
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

属性	说明
clocks	用于配置使用主控提供的 32k 时钟
pinctrl-0	用于配置 pin 的复用功能
pinctrl-names	用于配置 pin state
wlan_busnum	表示 WiFi 所使用的 SDIO 控制器号
wlan_power	表示给 WiFi 模组供电的 regulator 名称
wlan_io_regulator	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_regon	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_hostwake	表示 WiFi 唤醒主控的 GPIO
chip_en	表示 WiFi 模组使能引脚，硬件未使用时不配置
power_en	表示模块外部的电源开关控制引脚

说明

以上所有项必须参看原理图进行配置，配置与原理图实际使用的资源保持一致；最好是和硬件同事一起确认，比如有些设计供电可能是没有 axp 的，硬件直接供电了，所以不需要配置，特别注意的就是 sdio 的配置。

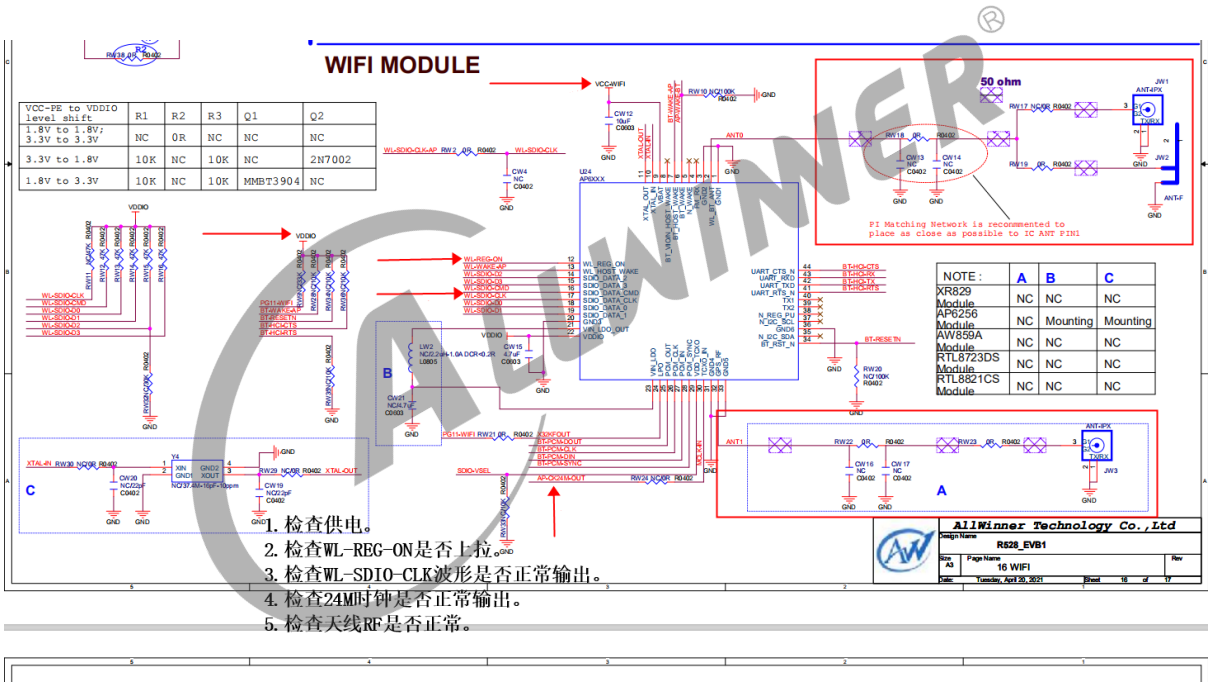


图 3-4: Tina-XR829 模组原理图

3.1.1.3 方案 module 适配

在 Tina4.0/target/allwinner/r528-common/modules.mk 中添加模块配置。

```
define KernelPackage/net-xr829-40M
 SUBMENU:=$(WIRELESS_MENU)
 TITLE:=xr829 support (staging)
 DEPENDS:= +xr829-firmware +@IPV6 +@XR829_USE_40M_SDD +@USES_XRADIO +@PACKAGE_xr829-rftest
 KCONFIG:=\
    CONFIG_XR829_WLAN=m \
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    CONFIG_PM=y\
    CONFIG_BT=y \
    CONFIG_BT_BREDR=y \
    CONFIG_BT_RFCOMM=y \
    CONFIG_BT_RFCOMM_TTY=y \
    CONFIG_BT_DEBUGFS=y \
    CONFIG_XR_BT_LPM=y \
    CONFIG_XR_BT_FDI=y \
    CONFIG_BT_HCIUART=y \
    CONFIG_BT_HCIUART_H4=y \
    CONFIG_HFP_OVER_PCM=y \
    CONFIG_RFKILL=y \
    CONFIG_RFKILL_PM=y \
    CONFIG_RFKILL_GPIO=y

 #FILES:=$(LINUX_DIR)/drivers/net/wireless/xr829/wlan/xradio_core.ko
 #FILES+=$(LINUX_DIR)/drivers/net/wireless/xr829/wlan/xradio_wlan.ko
 #FILES+=$(LINUX_DIR)/drivers/net/wireless/xr829/umac/xradio_mac.ko
 #AUTOLOAD:=$(call AutoProbe, xradio_mac xradio_core xradio_wlan)

 FILES:=$(LINUX_DIR)/drivers/net/wireless/xr829/xr829.ko
 AUTOLOAD:=$(call AutoProbe, xr829)
endef

define KernelPackage/net-xr829-40M/description
 Kernel modules for xr829 support
endef

$(eval $(call KernelPackage,net-xr829-40M))
```

几点说明：

- DEPENDS：Tina 包依赖配置
- KCONFIG：内核依赖配置
- FILES：内核模块路径

一般不需要用户自己适配，可以自己从已经适配过的方案的 modules.mk 中拷贝过来就好，也可以直接拷贝这段代码。

说明

XR829 分为 40M 和 24M 晶振，如果是 24M 晶振建议用下面的配置做区分。

```
define KernelPackage/net-xr829
 SUBMENU:=$(WIRELESS_MENU)
 TITLE:=xr829 support (staging)
 DEPENDS:= +xr829-firmware +@IPV6 +@USES_XRADIO +@PACKAGE_xr829-rftest +@PACKAGE_xr829-rftest
 KCONFIG:=\
    CONFIG_XR829_WLAN=m \
    CONFIG_PM=y\
    CONFIG_BT=y \
    CONFIG_BT_BREDR=y \
    CONFIG_BT_RFCOMM=y \
    CONFIG_BT_RFCOMM_TTY=y \
    CONFIG_BT_DEBUGFS=y \
    CONFIG_XR_BT_LPM=y \
    CONFIG_XR_BT_FDI=y \
    CONFIG_BT_HCIUART=y \
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    CONFIG_BT_HCIUART_H4=y \
    CONFIG_HFP_OVER_PCM=y \
    CONFIG_RFKILL=y \
    CONFIG_RFKILL_PM=y \
    CONFIG_RFKILL_GPIO=y

 #FILES:=$(LINUX_DIR)/drivers/net/wireless/xr829/wlan/xradio_core.ko
 #FILES+=$(LINUX_DIR)/drivers/net/wireless/xr829/wlan/xradio_wlan.ko
 #FILES+=$(LINUX_DIR)/drivers/net/wireless/xr829/umac/xradio_mac.ko
 #AUTOLOAD:=$(call AutoProbe, xradio_mac xradio_core xradio_wlan)

 FILES+=$(LINUX_DIR)/drivers/net/wireless/xr829/xr829.ko
 AUTOLOAD:=$(call AutoProbe, xr829)
endef

define KernelPackage/net-xr829/description
 Kernel modules for xr829 support
endef

$(eval $(call KernelPackage,net-xr829))
```

接着在 Tina4.0 根目录执行 m menuconfig 就可以看到新添加的模组。

```
make menuconfig
> Kernel modules > Wireless Drivers
```

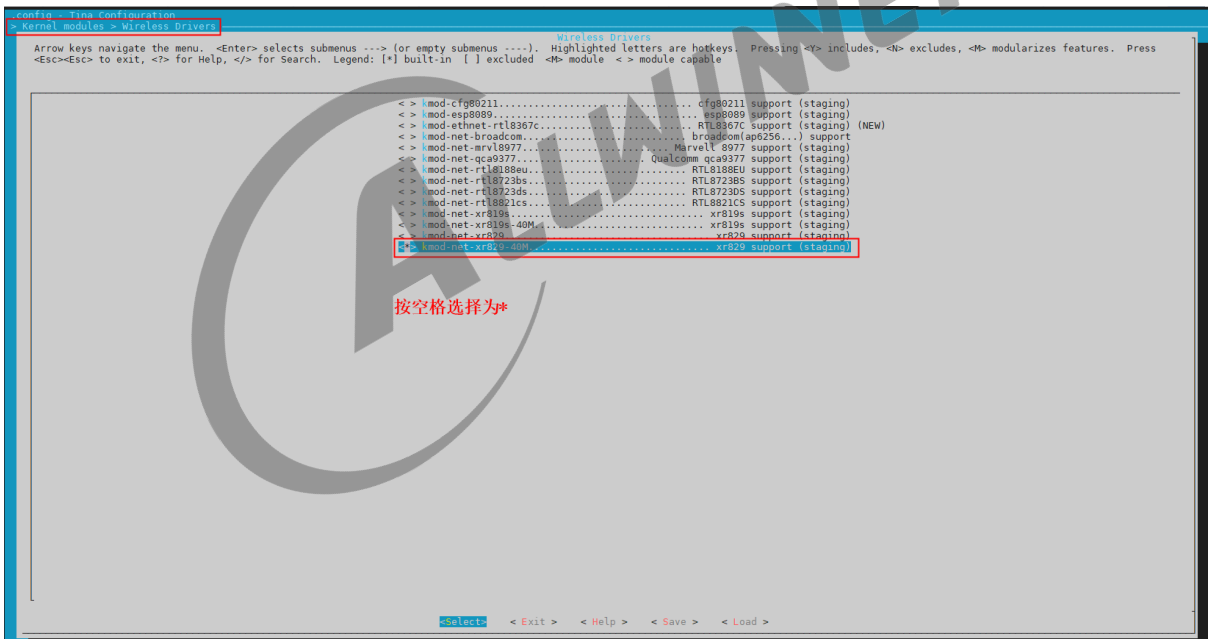


图 3-5: Tina-XR829-module 配置

3.1.1.4 添加 Firmware

在 Tina4.0/package/firmware/linux-firmware/添加 XR829 需要的 firmware。

```
Tina4.0/package/firmware/linux-firmwarexr829/
├──── boot_xr829.bin          //启动相关bin
├──── etf_xr829.bin           //etf测试工具相关bin
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
├──── fw_xr829_40M.bin          //wifi固件40M晶振需要的bin
├──── fw_xr829.bin              //wifi固件24M晶振需要的bin
├──── fw_xr829_bt_40M.bin       //bt固件40M晶振需要的bin
├──── fw_xr829_bt.bin           //bt固件24M晶振需要的bin
├──── sdd_xr829_40M.bin         //功率配置40M晶振的bin
├──── sdd_xr829.bin             //功率配置24M晶振的bin
└──── xr829.mk                  //Makefile文件
```

说明：功能调试阶段 wifi 只需要关注 boot_xr829.bin，fw_xr829.bin,sdd_xr829.bin。这些文件的获取一般伴随 XR829 产品包一起释放。

说明

一定要区分 24M 晶振和 40M 晶振。

Tina4.0 对应 xr829.mk 文件如下：

```
Package/xr829-firmware = $(call Package/firmware-default,Xradio xr829 firmware)

PKG_CONFIG_DEPENDS +=CONFIG_XR829_USE_40M_SDD

define Package/xr829-firmware/install
    $(INSTALL_DIR) $(1)/$(FIRMWARE_PATH)
    $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/xr829/boot_xr829.bin $(1)/$(FIRMWARE_PATH)/
      boot_xr829.bin
    $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/xr829/etf_xr829.bin $(1)/$(FIRMWARE_PATH)/
      etf_xr829.bin
    $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/xr829/fw_xr829.bin $(1)/$(FIRMWARE_PATH)/
      fw_xr829.bin
ifeq ($(CONFIG_XR829_USE_40M_SDD), y)
    $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/xr829/sdd_xr829_40M.bin
          $(1)/$(FIRMWARE_PATH)/sdd_xr829.bin
    $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/xr829/fw_xr829_bt_40M.bin
          $(1)/$(FIRMWARE_PATH)/fw_xr829_bt.bin
else
    $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/xr829/sdd_xr829.bin $(1)/$(FIRMWARE_PATH)/
      sdd_xr829.bin
    $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/xr829/fw_xr829_bt.bin
          $(1)/$(FIRMWARE_PATH)/fw_xr829_bt.bin
endif
endef
$(eval $(call BuildPackage,xr829-firmware))
```

Tina5.0/openwrt/openwrt/package/firmware/linux-firmware/xr829.mk 内容如下：

```
Package/xr829-firmware = $(call Package/firmware-default,Xradio xr829 firmware)

SRC_CODE_DIR:=$(XR829_FW)

define Package/xr829-firmware/install
    $(INSTALL_DIR) $(1)/$(FIRMWARE_PATH)
    $(INSTALL_DATA) $(SRC_CODE_DIR)/boot_xr829.bin $(1)/$(FIRMWARE_PATH)/boot_xr829.bin
    $(INSTALL_DATA) $(SRC_CODE_DIR)/etf_xr829.bin $(1)/$(FIRMWARE_PATH)/etf_xr829.bin
    $(INSTALL_DATA) $(SRC_CODE_DIR)/fw_xr829.bin $(1)/$(FIRMWARE_PATH)/fw_xr829.bin
ifeq ($(CONFIG_XR829_USE_40M_SDD), y)
    $(INSTALL_DATA) $(SRC_CODE_DIR)/sdd_xr829_40M.bin $(1)/$(FIRMWARE_PATH)/sdd_xr829.bin
    $(INSTALL_DATA) $(SRC_CODE_DIR)/fw_xr829_bt_40M.bin $(1)/$(FIRMWARE_PATH)/fw_xr829_bt.bin
else
    $(INSTALL_DATA) $(SRC_CODE_DIR)/sdd_xr829.bin $(1)/$(FIRMWARE_PATH)/sdd_xr829.bin
    $(INSTALL_DATA) $(SRC_CODE_DIR)/fw_xr829_bt.bin $(1)/$(FIRMWARE_PATH)/fw_xr829_bt.bin
endif
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
endef
$(eval $(call BuildPackage,xr829-firmware))
```

说明：可以看到整个 Makefile 文件就只是做了简单的拷贝动作。

说明

一般 firmware 在开发板上的路径是：系统的/lib/firmware/, 如果更改请确保是否和驱动中定义的保持一致，最新版驱动已经自适应寻找路径了, 早期的驱动版本一定要留意。

驱动中定义 firmware 文件为：

```
xr829/wlan/etf.h
#ifdef USE_VFS_FIRMWARE
#define XR829_ETF_FIRMWARE   ("/system/vendor/etc/firmware/etf_xr829.bin")
#else
#define XR829_ETF_FIRMWARE   ("etf_xr829.bin")
#endif

xr829/wlan/fwio.h
#ifdef USE_VFS_FIRMWARE
#define XR829_BOOTLOADER    ("/system/vendor/etc/firmware/boot_xr829.bin")
#define XR829_FIRMWARE      ("/system/vendor/etc/firmware/fw_xr829.bin")
#define XR829_SDD_FILE      ("/system/vendor/etc/firmware/sdd_xr829.bin")
#else
#define XR829_BOOTLOADER    ("boot_xr829.bin")
#define XR829_FIRMWARE      ("fw_xr829.bin")
#define XR829_SDD_FILE      ("sdd_xr829.bin")
#endif
```

接着在 Tina4.0 根目录执行 m menuconfig 就可以看到新添加的 firmware 配置。

```
> Firmware
  -*- xr829-firmware..... Xradio xr829 firmware
  (/lib/firmware/) Firmware's directory    //如想临时配置修改也可以改这里的路径
  -*- xr829 with 40M sdd
```

说明

晶振 24M 和 40M 的区分，根据实际硬件选择。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

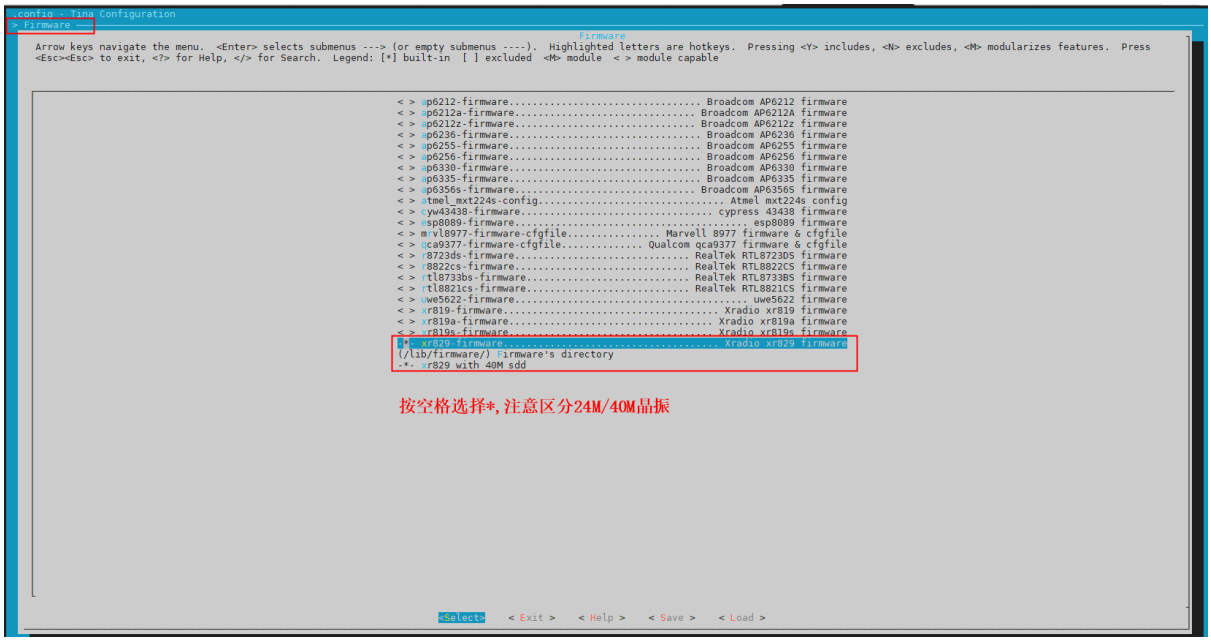


图 3-6: Tina-XR829-firmware 配置

3.1.1.5 应用工具适配

1. wifimanager 配置

```
make menuconfig
> Allwinner > wifimanager
```

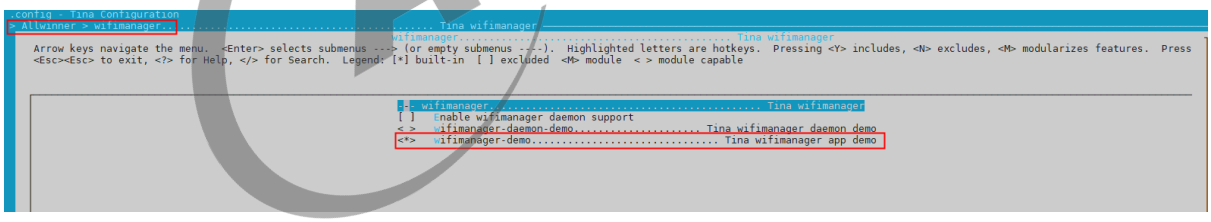


图 3-7: Tina-wifimanager 配置

2. smartlinkd 配网配置

```
make menuconfig
> Allwinner > smartlinkd
```

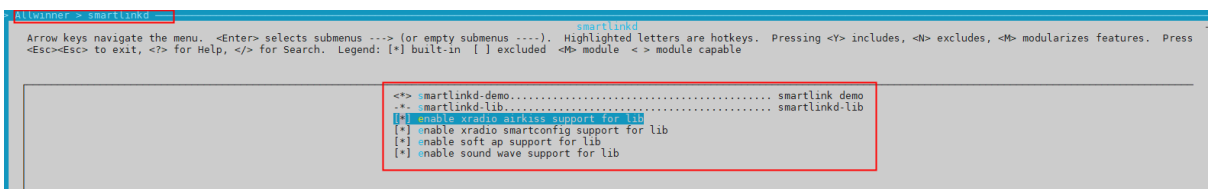


图 3-8: Tina-smartlinkd 配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3. softap 配置

```
make menuconfig
> Allwinner > softap
```

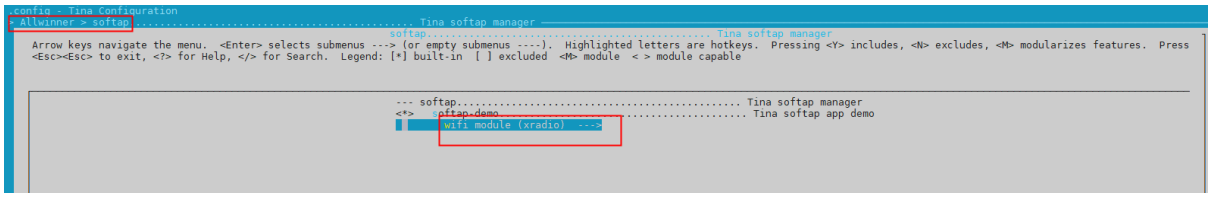


图 3-9: Tina-softap 配置

4. rf 工具配置

在 Tina4.0/package/utils/rftest 目录下添加 XR829 的 RF 测试工具（一般都会随产品包释放）。

```
tina/package/utils/rftest/xr829/
└──── etf
   ├──── btetf_1.4.0
   ├──── btetf_riscv_1.0.5
   ├──── etf
   ├──── etf_riscv_1.4.0
   └──── xrbt_testmode_1.4.0
```

在 Tina4.0/package/utils/rftest/Makefile 中添加。

```
define Package/xr829-rftest
 $(Package/$(PKG_NAME)/Default)
 TITLE:=xr829 rf test tools
endef
define Package/xr829-rftest/description
 $(call Package/$(PKG_NAME)/description/Default)
endef
define Package/xr829-rftest/install
	$(INSTALL_DIR) /$(1)/usr/bin
ifeq ($(TARGET_ARCH), riscv)
	$(INSTALL_BIN) ./xr829/etf/etf_riscv_1.4.0 /$(1)/usr/bin/etf
	$(INSTALL_BIN) ./xr829/etf/btetf_riscv_1.0.5 /$(1)/usr/bin/btetf
else
	$(INSTALL_BIN) ./xr829/etf/btetf_1.4.0 /$(1)/usr/bin/btetf
	$(INSTALL_BIN) ./xr829/etf/etf /$(1)/usr/bin/etf
	$(INSTALL_BIN) ./xr829/etf/xrbt_testmode_1.4.0 /$(1)/usr/bin/xrbt_testmode
endif
endef
$(eval $(call BuildPackage,xr829-rftest))
```

```
make menuconfig
> Utilities > rf test tool
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

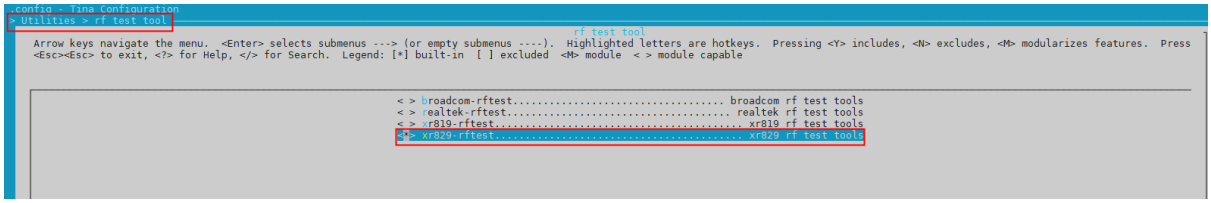


图 3-10: Tina-XRADIO-rf 工具配置

5. iperf 工具配置

```
make menuconfig
> Network
```

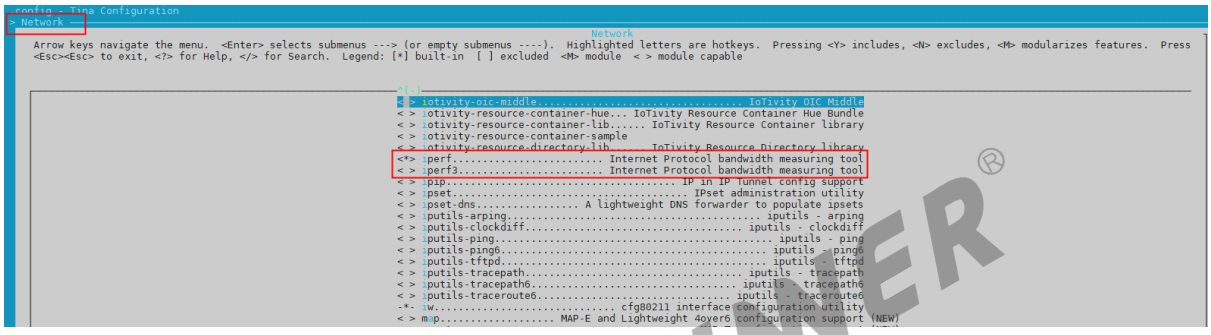


图 3-11: Tina-iperf 工具配置

6. wpa 工具配置

wpa 工具包括 wpa-supplicant 服务的和 wpa-cli 客户端。

```
make menuconfig
 > Network
```

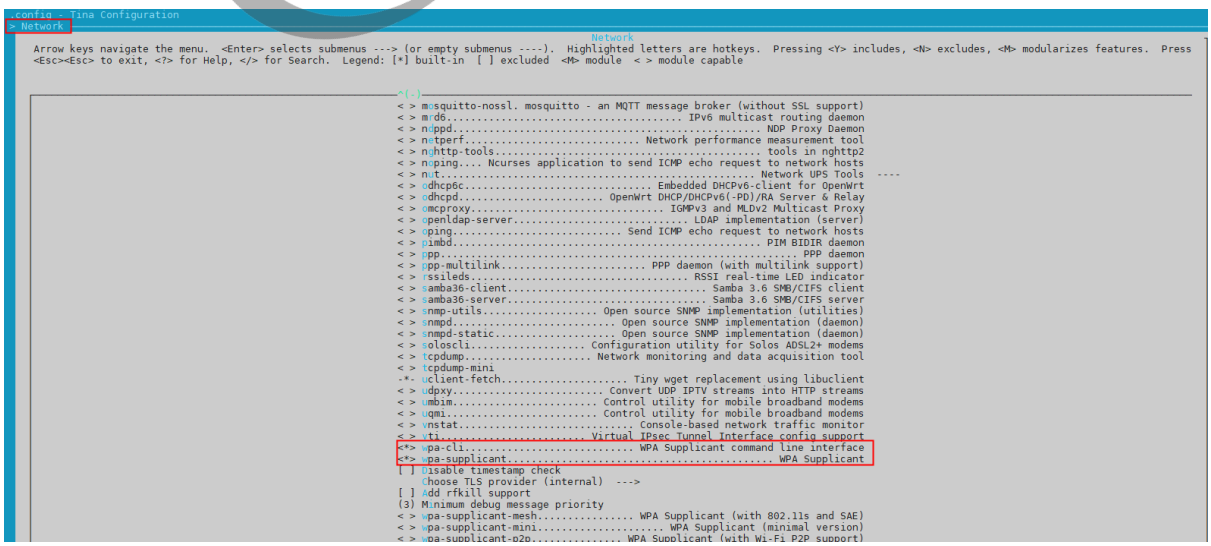


图 3-12: Tina-wpasupplicant 配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3.1.2 XR829 模组移植 (buildroot 编译系统)

主控：T113

无线模组：XR829

系统版本：Tina5.0 linux-5.4

方案：evb1_nand

3.1.2.1 内核驱动适配

内核驱动适配分为如下四个步骤：

- 获取源码。
- 添加 Kconfig 和 Makefile 配置。
- make kernel_menuconfig 配置。
- 编译。

1）获取 XR829 驱动源码，放到内核驱动路径下。客户可以从一号通账号获取。

```
tina-5.0/kernel/linux-5.4/drivers/net/wireless/xr829/
├── include
├── Kconfig
├── Makefile
├── umac
└── wlan
```

2）内核添加 Kconfig 和 Makefile 的配置。

tina-5.0/kernel/linux-5.4/drivers/net/wireless/Kconfig 文件中引入 XR829 驱动的 Kconfig 配置。

```
source "drivers/net/wireless/xr829/Kconfig"
```

tina-5.0/kernel/linux-5.4/drivers/net/wireless/Makefile 文件中引入 XR829 驱动的 Makefile 配置。

```
obj-$(CONFIG_XR829_WLAN)  += xr829/
```

3）./build.sh menuconfig 配置

在 Tina5.0 根目录执行./build.sh menuconfig。

a. XR829 驱动的配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

> Device Drivers > Network device support > Wireless LAN
<M> XR829 WLAN support

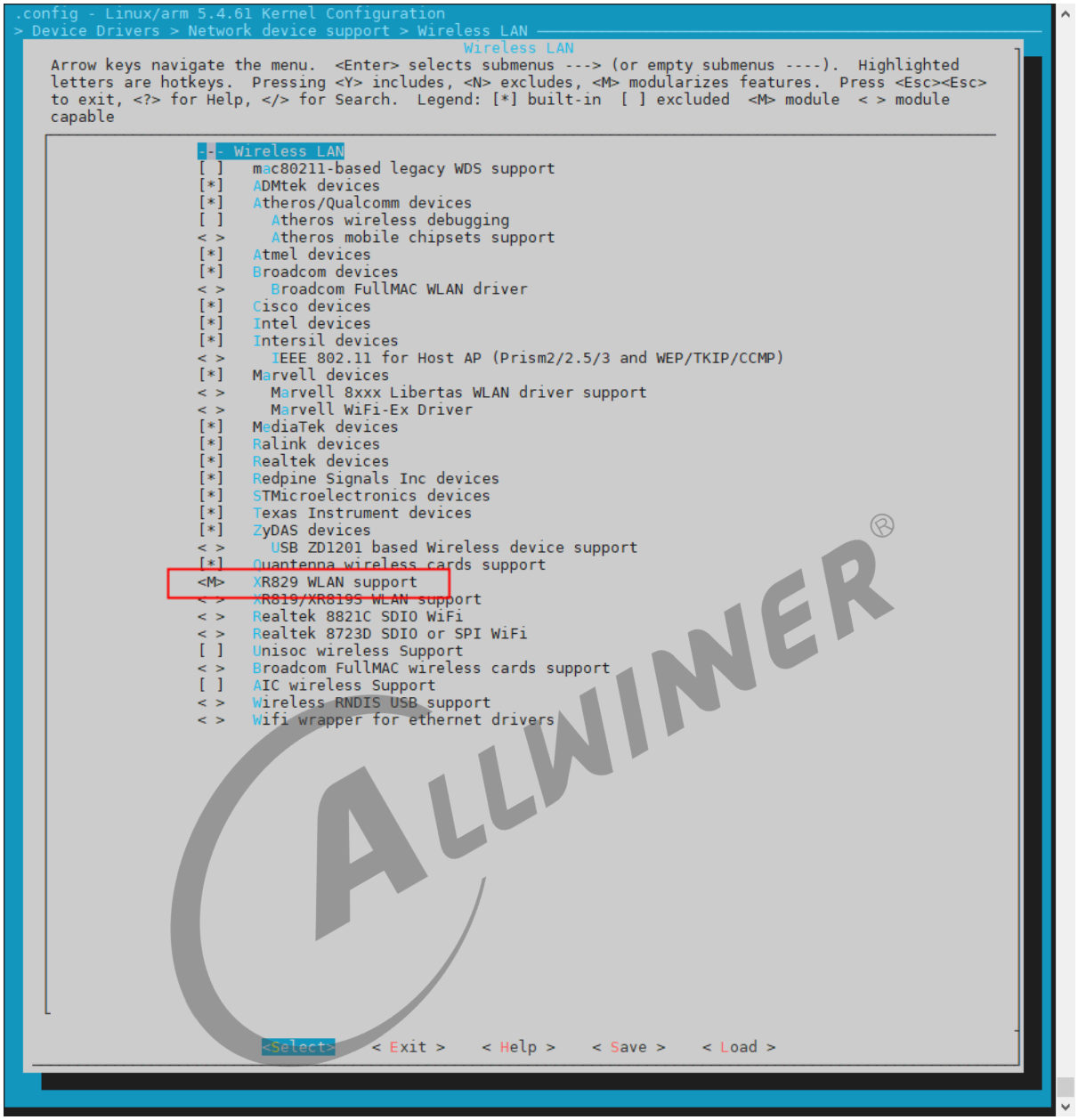


图 3-13: buildroot-XR829 内核配置

b. sunxi-rf 的配置

> Device Drivers > Misc devices
<*> Allwinner rfkill driver

版权所有 © 珠海全志科技股份有限公司。保留一切权利

图 3-14: Tina-buildroot-sunxi-rf 配置

c. SDIO 的配置

```
> Device Drivers > MMC/SD/SDIO card support
  <*>  Allwinner sunxi SD/MMC Host Controller support  --->
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

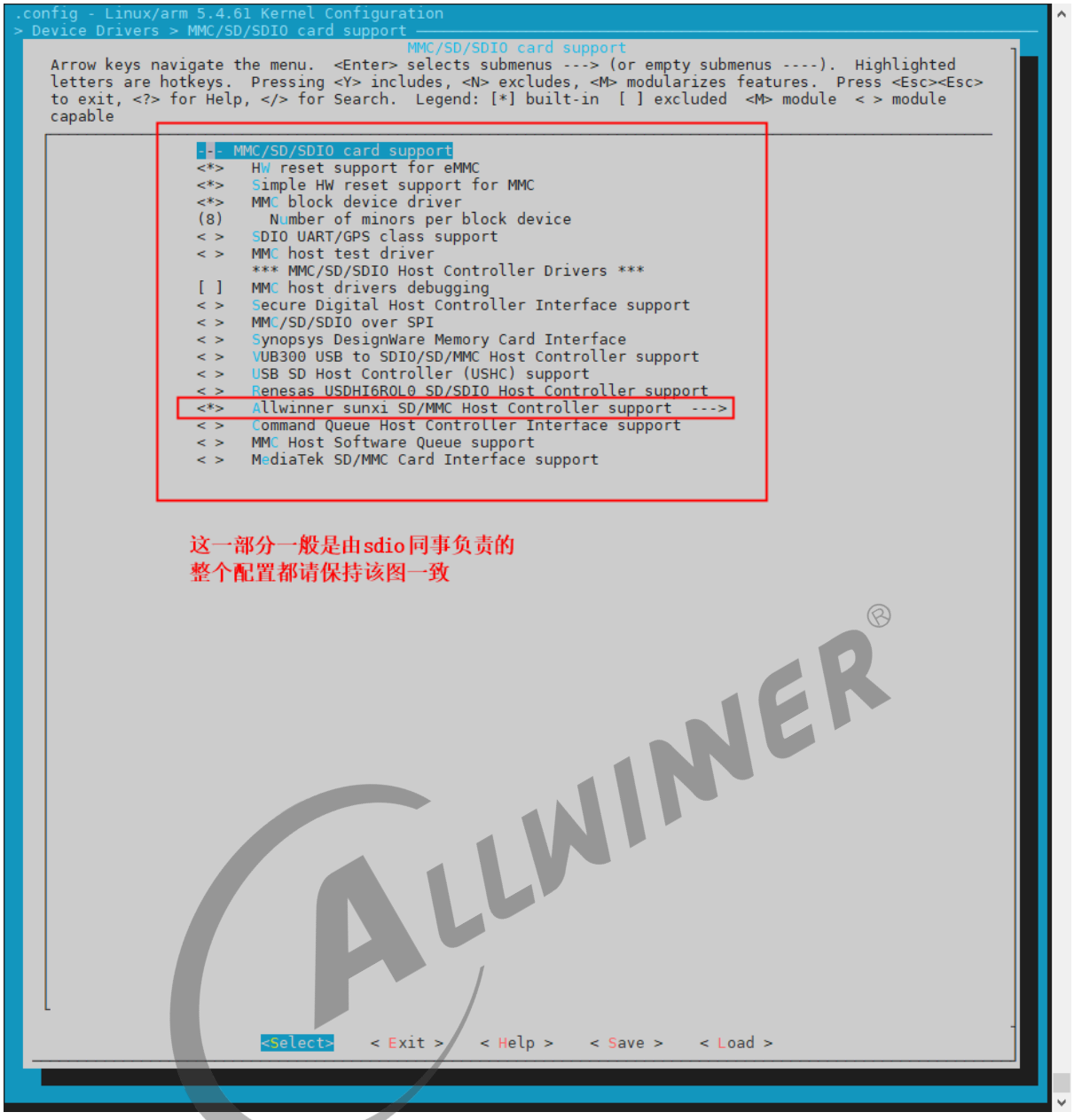


图 3-15: Tina-buildroot-sdc 配置

4) 编译

在 Tina5.0/根目录执行./build.sh kernel 进行编译。

```
Tina5.0/kernel/linux-5.4/drivers/net/wireless/xr829/
├──── include
├──── Kconfig
├──── Makefile
├──── modules.order
├──── umac
├──── wlan
├──── xr829.ko        //正常编译出xr829.ko模块
├──── xr829.mod
├──── xr829.mod.c
├──── xr829.mod.o
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
└──── xr829.o
```

3.1.2.2 硬件资源适配

在 tina-5.0/device/config/chips/t113_s3p/configs/evb1_nand 中添加引脚配置。

```
&soc {
rfkill: rfkill@0 {
        compatible   = "allwinner,sunxi-rfkill";
        chip_en;
        power_en;
        pinctrl-0 = <&wlan_pins_a>;
        pinctrl-names = "default";
        status       = "okay";

        wlan: wlan@0 {
            compatible   = "allwinner,sunxi-wlan";
            clock-names = "32k-fanout1";
            clocks = <&ccu CLK_FANOUT1_OUT>;
            wlan_busnum   = <0x1>;
            wlan_regon   = <&pio PE 3 GPIO_ACTIVE_HIGH>;
            wlan_hostwake  = <&pio PG 10 GPIO_ACTIVE_HIGH>;
            /*wlan_power   = "VCC-3V3";*/
            /*wlan_power_vol = <3300000>;*/
            /*interrupt-parent = <&pio>;
            interrupts = < PG 10 IRQ_TYPE_LEVEL_HIGH>;*/
            wakeup-source;

        };
.....
    };
};
```

属性	说明
clocks	用于配置使用主控提供的 32k 时钟
pinctrl-0	用于配置 pin 的复用功能
pinctrl-names	用于配置 pin state
wlan_busnum	表示 WiFi 所使用的 SDIO 控制器号
wlan_power	表示给 WiFi 模组供电的 regulator 名称
wlan_regon	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_hostwake	表示 WiFi 唤醒主控的 GPIO
chip_en	表示 WiFi 模组使能引脚，硬件未使用时不配置
power_en	表示模块外部的电源开关控制引脚

说明

以上所有项必须参看原理图进行配置，配置与原理图实际使用的资源保持一致；最好是和硬件同事一起确认，比如有些设计供电可能是没有 axp 的，硬件直接供电了，所以不需要配置，特别注意的就是 sdio 的配置。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

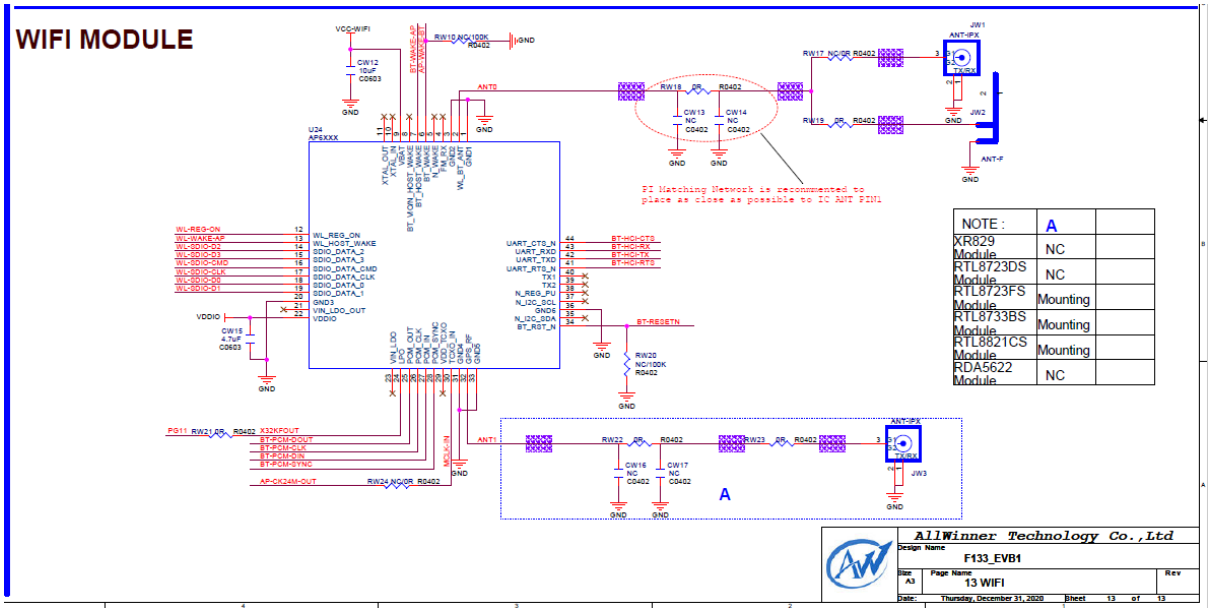


图 3-16: Tina-buildroot-XR829 模组原理图

3.1.2.3 方案 module 适配

说明

buildroot 编译方式无 module 适配，系统起来后需手动加载模组或根据实际情况自编写启动自加载脚本。

3.1.2.4 添加 Firmware

在 tina-5.0/platform/allwinner/wireless/firmware 添加 XR829 需要的 firmware。

```
Tina4.0/package/firmware/linux-firmwarexr829/
 ├──── boot_xr829.bin          //启动相关bin
 ├──── etf_xr829.bin           //etf测试工具相关bin
 ├──── fw_xr829_40M.bin        //wifi固件40M晶振需要的bin
 ├──── fw_xr829.bin            //wifi固件24M晶振需要的bin
 ├──── fw_xr829_bt_40M.bin     //bt固件40M晶振需要的bin
 ├──── fw_xr829_bt.bin         //bt固件24M晶振需要的bin
 ├──── sdd_xr829_40M.bin       //功率配置40M晶振的bin
 ├──── sdd_xr829.bin           //功率配置24M晶振的bin
```

说明：功能调试阶段 wifi 只需要关注 boot_xr829.bin，fw_xr829.bin,sdd_xr829.bin。这些文件的获取一般伴随 XR829 产品包一起释放。

说明

一定要区分 24M 晶振和 40M 晶振。

修改 tina-5.0/buildroot/config/buildroot/wifi-firmware/wifi-firmware.mk 文件

```
define WIFI_FIRMWARE_INSTALL_TARGET_CMDS
    $(INSTALL) -d -m 0755 $(FIRMWARE_DIR)/
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
.....
    if test "$(BR2_PACKAGE_XR829_USE_40M)" = "y" ; then \
        cp -r $(WIFI_FIRMWARE_SITE)/xr829/* $(FIRMWARE_DIR) ; \
        mv $(FIRMWARE_DIR)/fw_xr829_bt_40M.bin $(FIRMWARE_DIR)/fw_xr829_bt.bin ; \
        mv $(FIRMWARE_DIR)/sdd_xr829_40M.bin $(FIRMWARE_DIR)/sdd_xr829.bin ; \
    fi;

    if test "$(BR2_PACKAGE_XR829_USE_24M)" = "y" ; then \
        cp -r $(WIFI_FIRMWARE_SITE)/xr829/* $(FIRMWARE_DIR) ; \
    fi;
.....
endef

$(eval $(generic-package))
```

说明：可以看到整个 Makefile 文件就只是做了简单的拷贝动作。

说明

一般 firmware 的路径是：系统的/lib/firmware/, 如果更改请确保是否和驱动中定义的保持一致，最新版驱动已经自适应寻找路径了, 早期的驱动版本一定要留意。

驱动中定义 firmware 文件为：

```
xr829/wlan/etf.h
#ifdef USE_VFS_FIRMWARE
#define XR829_ETF_FIRMWARE   ("/system/vendor/etc/firmware/etf_xr829.bin")
#else
#define XR829_ETF_FIRMWARE   ("etf_xr829.bin")
#endif

xr829/wlan/fwio.h
#ifdef USE_VFS_FIRMWARE
#define XR829_BOOTLOADER   ("/system/vendor/etc/firmware/boot_xr829.bin")
#define XR829_FIRMWARE     ("/system/vendor/etc/firmware/fw_xr829.bin")
#define XR829_SDD_FILE     ("/system/vendor/etc/firmware/sdd_xr829.bin")
#else
#define XR829_BOOTLOADER   ("boot_xr829.bin")
#define XR829_FIRMWARE     ("fw_xr829.bin")
#define XR829_SDD_FILE     ("sdd_xr829.bin")
#endif
```

修改 tina-5.0/buildroot/config/buildroot/wifi-firmware/Config.in 文件，添加可选配置项

```
config BR2_PACKAGE_XR829_FIRMWARE
    bool "xr829-firmware"
    help
     xr829 firmware.

config BR2_PACKAGE_XR829_USE_40M
    bool "xr829_40M"
    depends on BR2_PACKAGE_XR829_FIRMWARE
    help
     xr829 40M crystal oscillator.

config BR2_PACKAGE_XR829_USE_24M
    bool "xr829_24M"
    depends on BR2_PACKAGE_XR829_FIRMWARE
    help
     xr829 24M crystal oscillator.
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

接着在 Tina5.0 根目录执行./build.sh buildroot_menuconfig 就可以看到新添加的 firmware 配置。

说明

晶振 24M 和 40M 的区分，根据实际硬件选择。

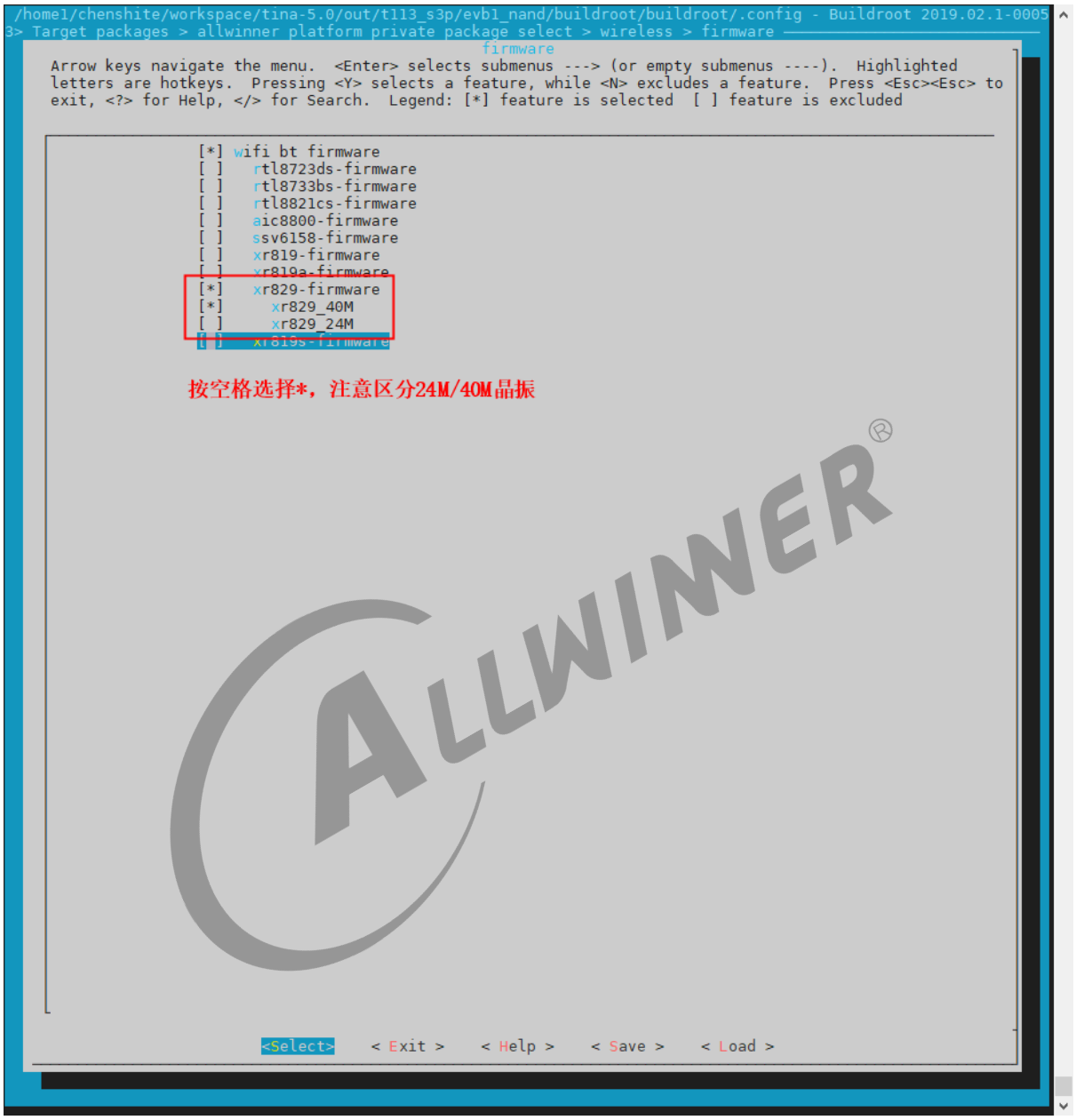


图 3-17: Tina-buildroot-XR829-firmware 配置

3.1.2.5 应用工具适配

1. wifimanager 配置 (buildroot 编译方式暂时不支持 wifimanager 定制化配置)

```
./build.sh buildroot_menuconfig
> Target packages > allwinner platform private package select > wireless
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

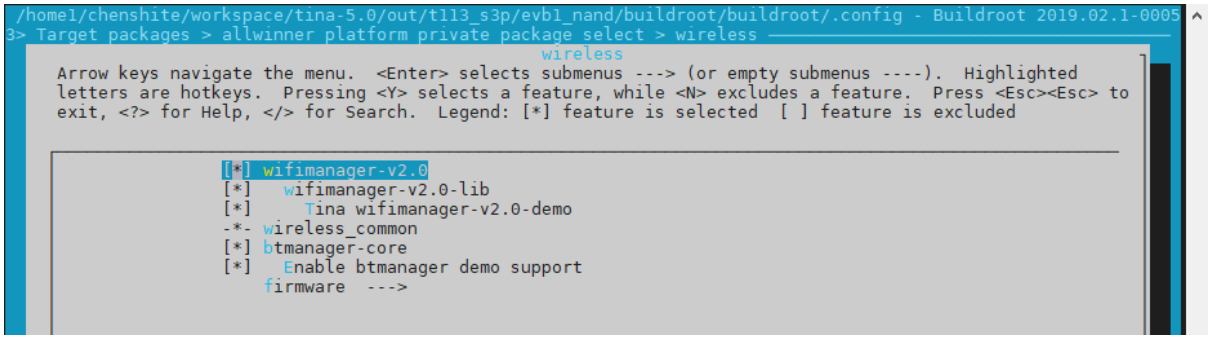


图 3-18: Tina-buildroot-wifimanager 配置

2. rf 工具配置 (buildroot 编译方式暂不支持 rf 工具)

3. iperf 工具配置

```
./build.sh buildroot_menuconfig
> Target packages > Networking applications
```

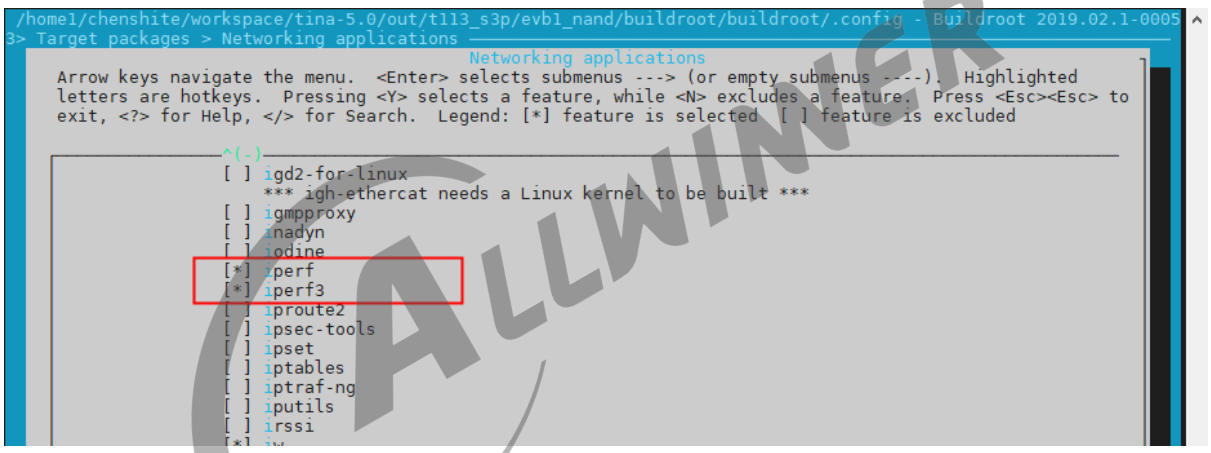


图 3-19: Tina-buildroot-iperf 工具配置

4. wpa/hostapd 工具配置

wpa 工具包括 wpa-supplicant 服务的和 wpa-cli 客户端。

```
./build.sh buildroot_menuconfig
> Target packages > Networking applications
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

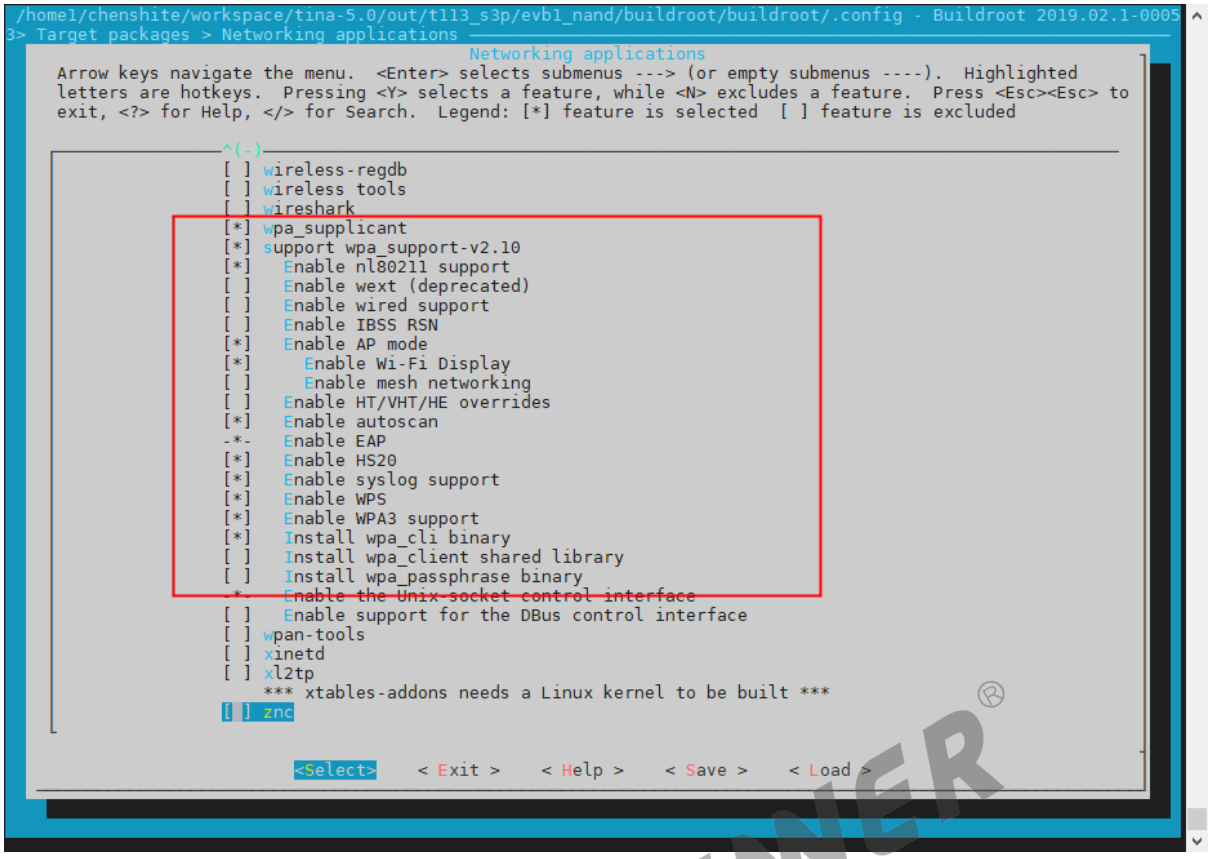


图 3-20: Tina-buildroot-wpasupplicant 配置

hostapd 工具。

```
./build.sh buildroot_menuconfig
> Target packages > Networking applications
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

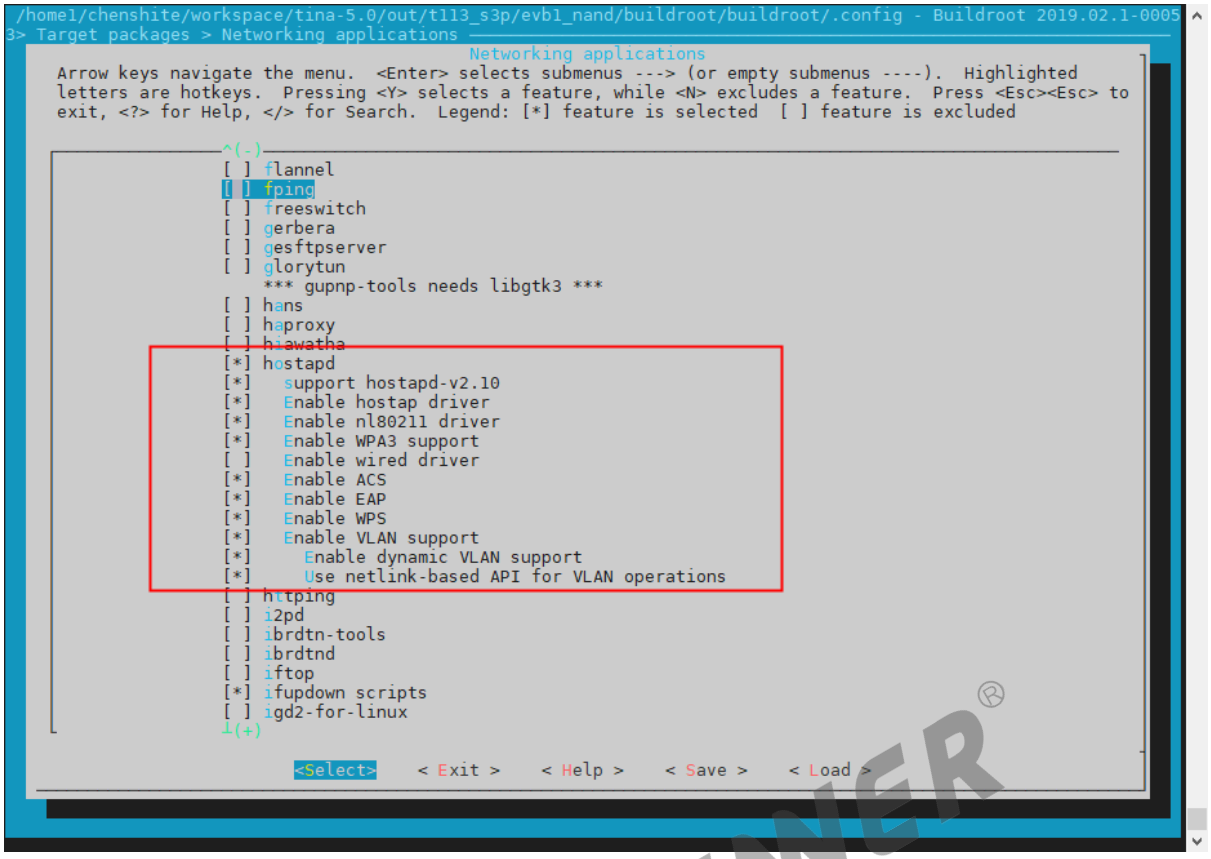


图 3-21: Tina-buildroot-hostapd 配置

3.2 瑞昱 RTL 系列

瑞昱的无线模组型号多种多样，目前我司搭配的方案主要有:

表 3-4: RTL 支持列表

Vendor	PartNumber	Interface	Wi-Fi	BT
Realtek	rtl8723bs	SDIO+UART	b/g/n	支持
Realtek	rtl8723ds	SDIO+UART	b/g/n	支持
Realtek	rtl8723cs	SDIO+UART	b/g/n	支持
Realtek	rtl8733bs	SDIO+UART	b/g/n	支持
Realtek	rtl8821cs	SDIO+UART	b/g/n/ac	支持
Realtek	rtl8822cs	SDIO+UART	b/g/n/ac	支持
Realtek	rtl8189fs	SDIO+UART	b/g/n	支持
Realtek	rtl8188fu	SDIO+UART	b/g/n	支持

本系列将以 RTL8723DS 和 RTL8821CS 两个典型模组为例详细介绍移植操作。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3.2.1 RTL8723DS 模组移植

主控：R528

无线模组：RTL8723DS

内核版本：linux-5.4

方案：r528_evb1-tina

3.2.1.1 内核驱动适配

内核驱动适配分为如下几个步骤：

- 获取源码。
- 添加 Kconfig 和 Makefile 配置。
- 修改驱动源码
- make kernel_menuconfig 配置。
- 编译。

1）获取 RTL8723DS 驱动源码，放到内核驱动路径下。【请直接联系原厂提供】

```
Tina4.0/lichee/linux-5.4/drivers/net/wireless/rtl8723ds/
├── clean
├── core
├── hal
├── ifcfg-wlan0
├── include
├── Kconfig
├── Makefile
├── os_dep
├── platform
├── runwpa
└── wlan0dhcp
```

2）内核添加 Kconfig 和 Makefile 的配置。

Tina4.0/lichee/linux-5.4/drivers/net/wireless/Kconfig 文件中引入 RTL8723DS 驱动的 Kconfig 配置。

```
source "drivers/net/wireless/rtl8723ds/Kconfig"
```

Tina4.0/lichee/linux-5.4/drivers/net/wireless/Makefile 文件中引入 RTL8723DS 驱动的 Makefile 配置。

```
obj-$(CONFIG_RTL8723DS)   += rtl8723ds/
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3）修改驱动源码

Tina4.0/lichee/linux-5.4/drivers/net/wireless/rtl8723ds/修改驱动原生代码的 Makefile。

```
CONFIG_RTW_ANDROID = 0       //0: no Android, 4/5/6/7/8/9/10 : Android version
CONFIG_PLATFORM_I386_PC = n    //原厂默认提供的是针对i386的平台适配
CONFIG_PLATFORM_ARM_SUNxI = y  //平台宏适配改为allwinner
CONFIG_RTW_DEBUG = y         //debug宏控制，适配初期建议打开
CONFIG_RTW_LOG_LEVEL = 3     //debug等级调整，0-4,建议设置成3，确定是驱动的问题再设置为4跟踪
```

Tina4.0/lichee/linux-5.4/drivers/net/wireless/rtl8723ds/platform 替换平台文件，主要涉及上下电，扫卡等函数。

platform_ARM_SUNxI_sdio.c 平台文件可以从已经移植的 RTL 模组中替换到该路径，或者按照第二章说明逐步实现上电扫卡函数。

4）make kernel_menuconfig 配置

在 Tina4.0/lichee/linux-5.4 内核目录执行 m kernel_menuconfig

a. RTL8723DS 驱动配置

```
> Device Drivers > Network device support > Wireless LAN
    <M>   Realtek 8723D SDIO or SPI WiFi
```

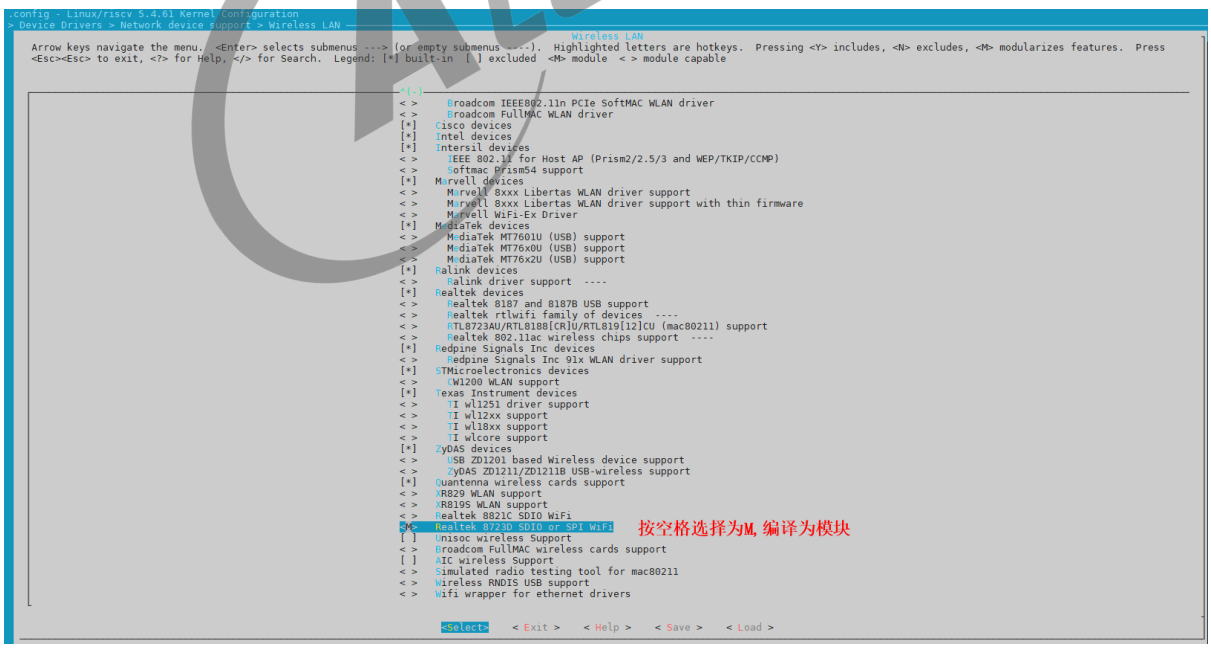


图 3-22: Tina-RTL8723DS 内核配置

b. sunxi-rf 配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

> Device Drivers > Misc devices
  <*> Allwinner rfkill driver

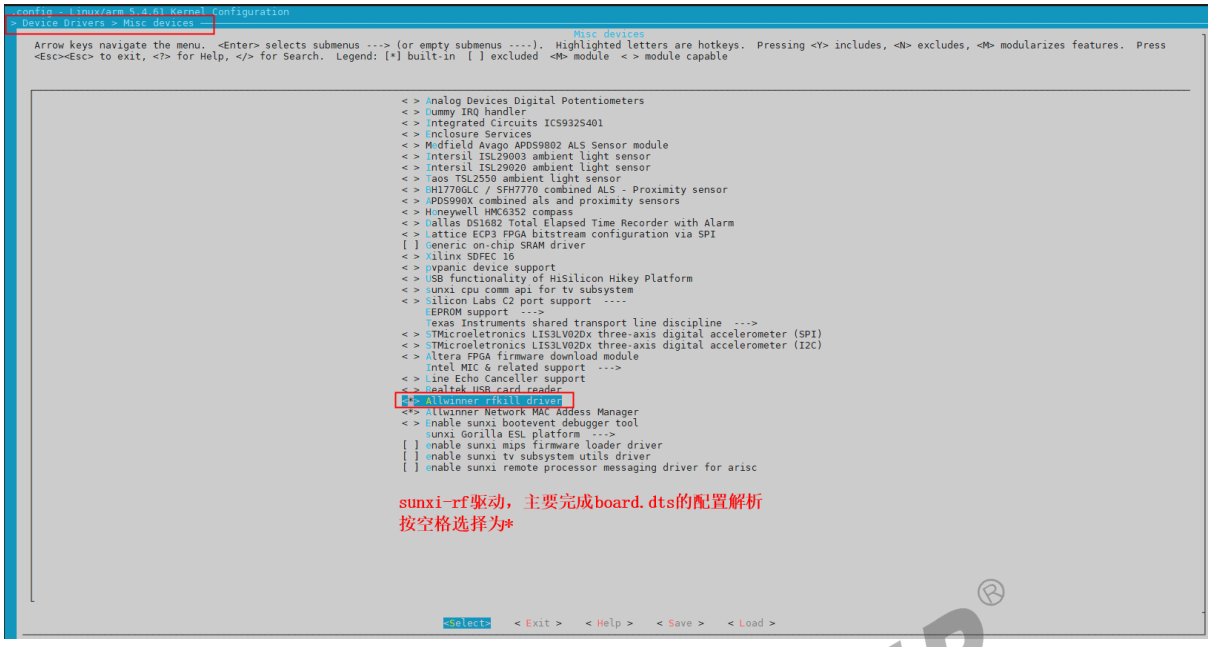


图 3-23: Tina-sunxi-rf 配置

c. SDIO 配置

> Device Drivers > MMC/SD/SDIO card support
  <*> Allwinner sunxi SD/MMC Host Controller support --->

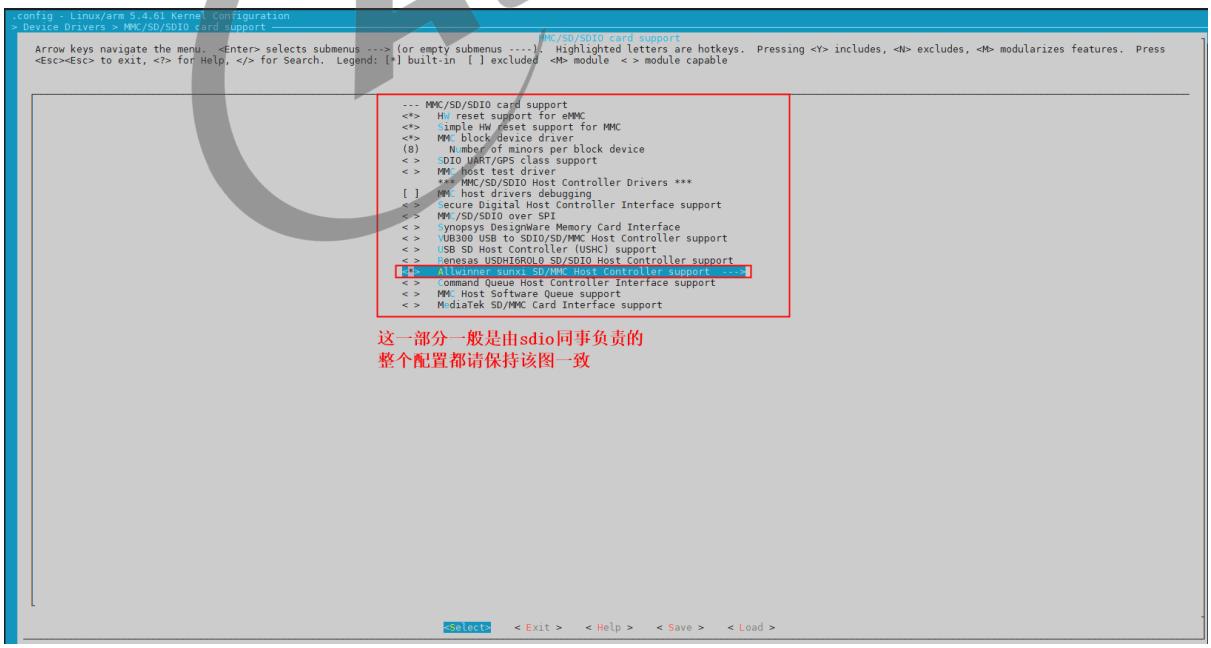


图 3-24: Tina-sdc 配置

5) 编译

版权所有 © 珠海全志科技股份有限公司。保留一切权利

在 Tina4.0/lichee/linux-5.4 内核目录执行 mkernel 进行编译。

Tina4.0/lichee/linux-5.4]$ mkernel

```
Tina4.0/lichee/linux-5.4/drivers/net/wireless/rtl8723ds/
├──── 8723ds.ko          //正常编译出8723ds.ko模块
├──── 8723ds.mod
├──── 8723ds.mod.c
├──── 8723ds.mod.o
├──── 8723ds.o
├──── clean
├──── core
├──── hal
├──── ifcfg-wlan0
├──── include
├──── Kconfig
├──── Makefile
├──── modules.order
├──── os_dep
├──── platform
├──── runwpa
└──── wlan0dhcp
```

3.2.1.2 硬件资源适配

在 Tina4.0/device/config/chips/r528/configs/evb1/linux/board.dts 中添加引脚配置。

```
&pio {
wlan_pins_a:wlan@0 {
        pins = "PG11";
        function = "clk_fanout1";  //这里涉及一个特殊处理，该方案的32k采用PG11复用给出
    };
...
};

&soc {
    rfkill: rfkill@0 {
        compatible   = "allwinner,sunxi-rfkill";
        chip_en;
        power_en;
        pinctrl-0 = <&wlan_pins_a>;
        pinctrl-names = "default";
        status      = "okay";

        wlan: wlan@0 {
            compatible   = "allwinner,sunxi-wlan";
            clock-names = "32k-fanout1";
            clocks = <&ccu CLK_FANOUT1_OUT>;
            wlan_busnum   = <0x1>;
            wlan_regon   = <&pio PE 17 GPIO_ACTIVE_HIGH>;
            wlan_hostwake  = <&pio PG 10 GPIO_ACTIVE_HIGH>;
            /*wlan_power   = "VCC-3V3";*/
            /*wlan_power_vol = <3300000>;*/
            /*interrupt-parent = <&pio>;
            interrupts = < PG 10 IRQ_TYPE_LEVEL_HIGH>;*/
            wakeup-source;
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
        };

...
    };
};
```

属性	说明
clocks	用于配置使用主控提供的 32k 时钟
pinctrl-0	用于配置 pin 的复用功能
pinctrl-names	用于配置 pin state
wlan_busnum	表示 WiFi 所使用的 SDIO 控制器号
wlan_power	表示给 WiFi 模组供电的 regulator 名称
wlan_io_regulator	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_regon	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_hostwake	表示 WiFi 唤醒主控的 GPIO
chip_en	表示 WiFi 模组使能引脚，硬件未使用时不配置
power_en	表示模块外部的电源开关控制引脚

说明

所有项必须参看原理图进行配置，配置与原理图实际使用的资源保持一致；最好是和硬件同事一起确认，比如有些设计供电可能是没有 axp 的，硬件直接供电了，所以不需要配置，特别注意的就是 sdio 的配置。

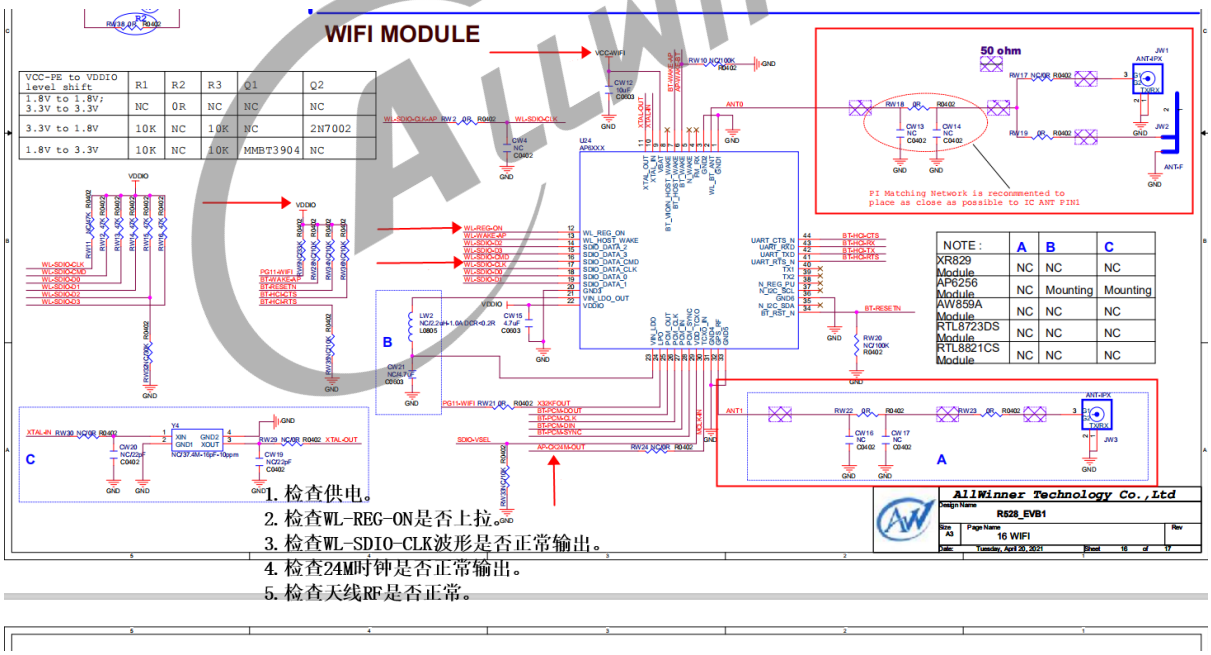


图 3-25: RTL8723DS 模组原理图

3.2.1.3 方案 module 适配

在 Tina4.0/target/allwinner/r528-common/modules.mk 中添加模块配置。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
define KernelPackage/net-rtl8723ds
 SUBMENU:=$(WIRELESS_MENU)
 TITLE:=RTL8723DS support (staging)
 DEPENDS:= +r8723ds-firmware +@IPV6 +@USES_REALTEK +@PACKAGE_realtek-rftest +@PACKAGE_rtk_hciattach
 FILES:=$(LINUX_DIR)/drivers/net/wireless/rtl8723ds/8723ds.ko
 KCONFIG:=\
     CONFIG_RTL8723DS=m \
     CONFIG_BT=y \
     CONFIG_BT_BREDR=y \
     CONFIG_BT_RFCOMM=y \
     CONFIG_BT_RFCOMM_TTY=y \
     CONFIG_BT_DEBUGFS=y \
     CONFIG_BT_HCIUART_RTL3WIRE=y \
     CONFIG_BT_HCIUART=y \
     CONFIG_BT_HCIUART_H4=y \
     CONFIG_HFP_OVER_PCM=y \
     CONFIG_RFKILL=y \
     CONFIG_RFKILL_PM=y \
     CONFIG_RFKILL_GPIO=y

 AUTOLOAD:=$(call AutoProbe,8723ds)
endef

define KernelPackage/net-rtl8723ds/description
 Kernel modules for RealTek RTL8723DS support
endef

$(eval $(call KernelPackage,net-rtl8723ds))
```

接着在 Tina4.0 根目录执行 m menuconfig 就可以看到新添加的模组。

```
> Kernel modules > Wireless Drivers
    <*> kmod-net-rtl8723ds..... RTL8723DS support (staging)
```

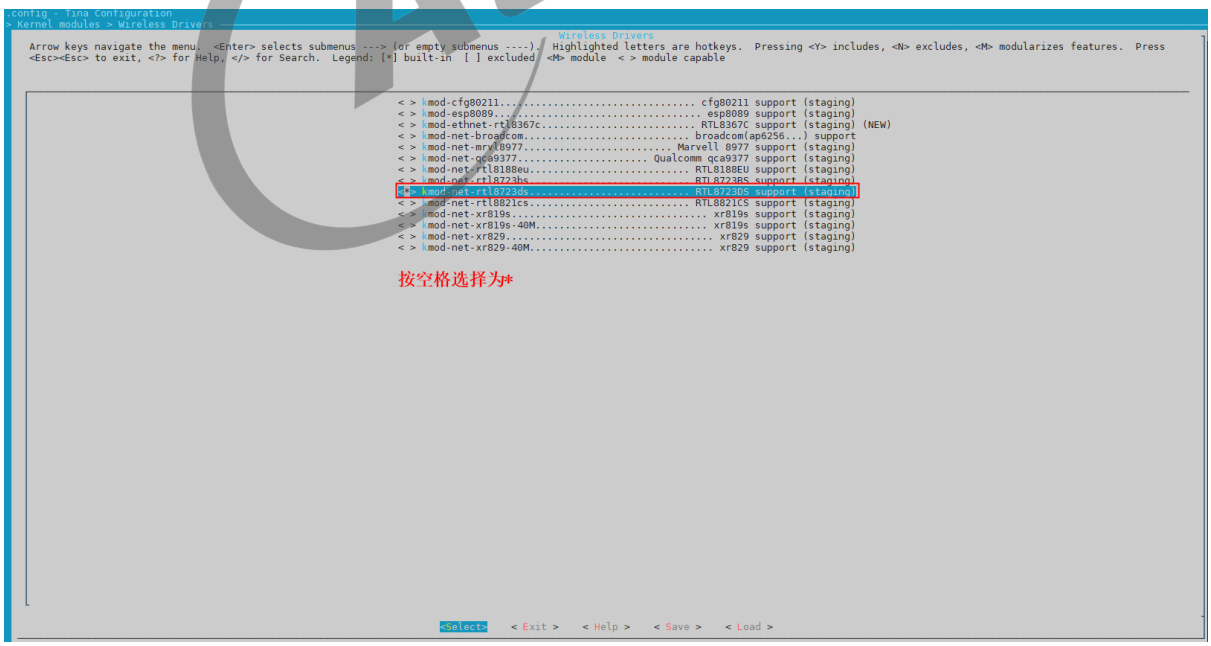


图 3-26: Tina-RTL8723DS-module 配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3.2.1.4 添加 Firmware

在 Tina4.0/package/firmware/linux-firmware/添加 rtl8723ds 需要的 firmware。

```
Tina4.0/package/firmware/linux-firmware/rtl8723ds/
├──── rtl8723d_config
├──── rtl8723d_fw
├──── rtl8723dsh4_config
├──── rtl8723dsh4_fw
└──── rtl8723ds.mk
```

rtl8723ds.mk 文件如下：

```
Package/r8723ds-firmware = $(call Package/firmware-default,RealTek RTL8723DS firmware)
define Package/r8723ds-firmware/install
  $(INSTALL_DIR) $(1)/$(FIRMWARE_PATH)/rtlbt
  $(INSTALL_DATA) \
    $(TOPDIR)/package/firmware/linux-firmware/rtl8723ds/rtl8723dsh4_fw \
    $(1)/$(FIRMWARE_PATH)/rtlbt/rtl8723dsh4_fw
  $(INSTALL_DATA) \
    $(TOPDIR)/package/firmware/linux-firmware/rtl8723ds/rtl8723dsh4_config \
    $(1)/$(FIRMWARE_PATH)/rtlbt/rtl8723dsh4_config
  cp \
    $(TOPDIR)/package/firmware/linux-firmware/rtl8723ds/rtl8723d_fw \
    $(1)/$(FIRMWARE_PATH)/rtlbt/rtl8723d_fw
  cp \
    $(TOPDIR)/package/firmware/linux-firmware/rtl8723ds/rtl8723d_config \
    $(1)/$(FIRMWARE_PATH)/rtlbt/rtl8723d_config
endef
$(eval $(call BuildPackage,r8723ds-firmware))
```

说明：可以看到整个 Makefile 文件就只是做了简单的拷贝动作。

```
make menuconfig
> Allwinner > smartlinkd
```

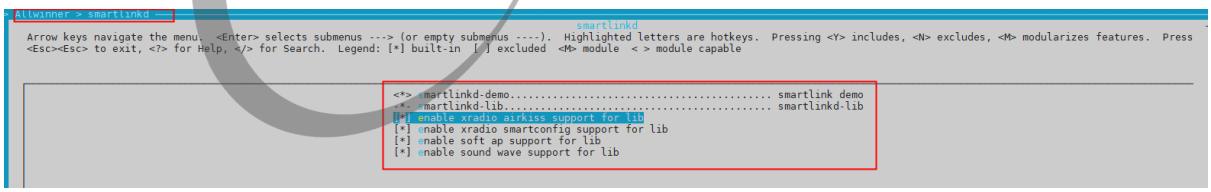


图 3-27: Tina-smartlinkd 配置

3. softap 配置

```
make menuconfig
> Allwinner > softap
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

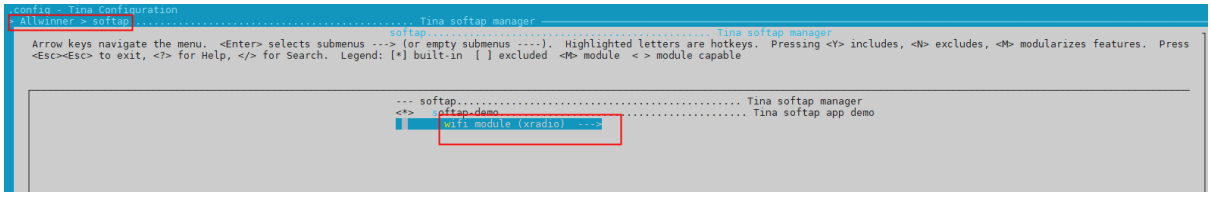


图 3-28: Tina-softap 配置

4. rf 工具配置

在 Tina4.0/package/utils/rftest 目录下添加 rtl8723ds 的 RF 测试工具（一般都会随产品包释放）。

```
tina/package/utils/rftest/realtek/
├── Android.mk
├── BT
├── build_platform
├── Makefile
├── readme.txt
├── rtw_api.cpp
├── rtw_api.h
├── rtw_Config.txt
├── rtwpriv.cpp
└── rtwpriv.h
```

在 Tina4.0/package/utils/rftest/Makefile 中添加。

```
define Package/realtek-rftest
  $(Package/$(PKG_NAME)/Default)
  TITLE:=realtek rf test tools
  DEPENDS:=+libcutils +liblog +libc +libstdcpp
endef
define Package/realtek-rftest/description
  $(call Package/$(PKG_NAME)/description/Default)
endef
define Package/realtek-rftest/install
	$(INSTALL_DIR) /$(1)/usr/sbin
	$(INSTALL_DIR) /$(1)/lib/firmware
	$(INSTALL_BIN) $(PKG_INSTALL_DIR)/rtwpriv $(1)/usr/sbin/rtwpriv
ifeq ($(findstring aarch64, $(TARGET_CC)), aarch64)
	$(INSTALL_BIN) ./realtek/BT/rtlbtmp_64 $(1)/usr/sbin/rtlbtmp
	$(INSTALL_DATA) ./realtek/BT/mp_rtl8723ds_config_64 $(1)/lib/firmware/mp_rtl8723ds_config
	$(INSTALL_DATA) ./realtek/BT/mp_rtl8723d_fw_64 $(1)/lib/firmware/mp_rtl8723d_fw
else
	$(INSTALL_BIN) ./realtek/BT/rtlbtmp $(1)/usr/sbin/rtlbtmp
	$(INSTALL_DATA) ./realtek/BT/mp_rtl8723d_config_h5 $(1)/lib/firmware/mp_rtl8723d_config_h5
	$(INSTALL_DATA) ./realtek/BT/mp_rtl8723d_config_h4 $(1)/lib/firmware/mp_rtl8723d_config_h4
	$(INSTALL_DATA) ./realtek/BT/mp_rtl8723d_fw $(1)/lib/firmware/mp_rtl8723d_fw
endif
endef
$(eval $(call BuildPackage,realtek-rftest))
```

```
make menuconfig
> Utilities > rf test tool
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

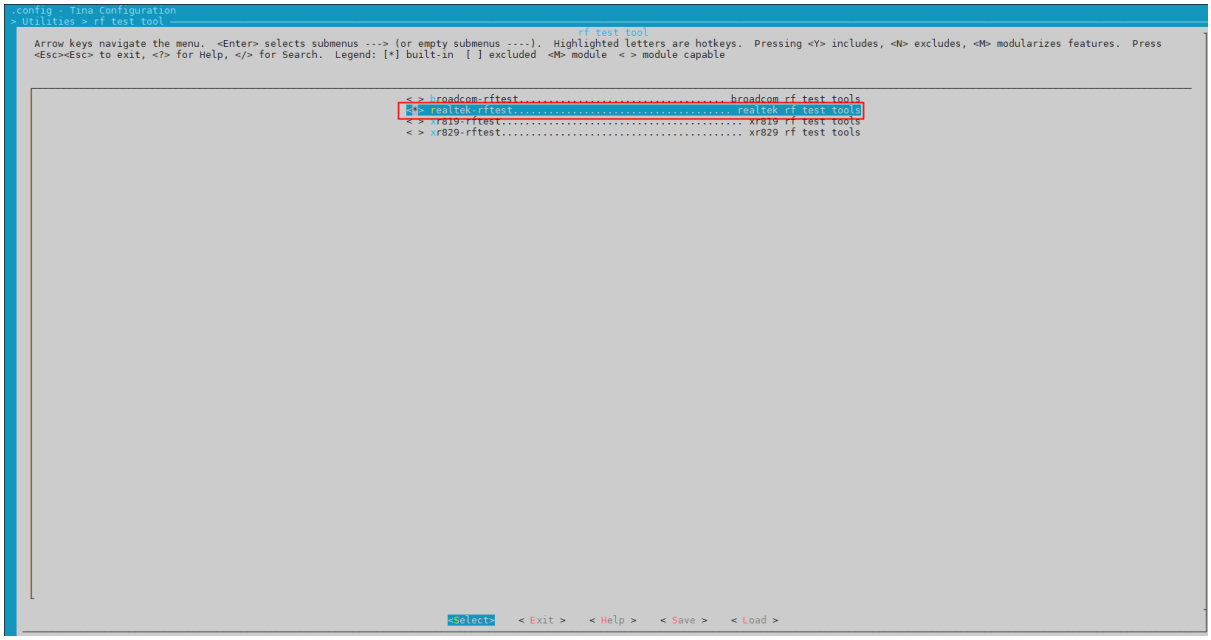


图 3-29: Tina-RTL-rf 工具配置

5. iperf 工具配置

```
make menuconfig
> Network
```

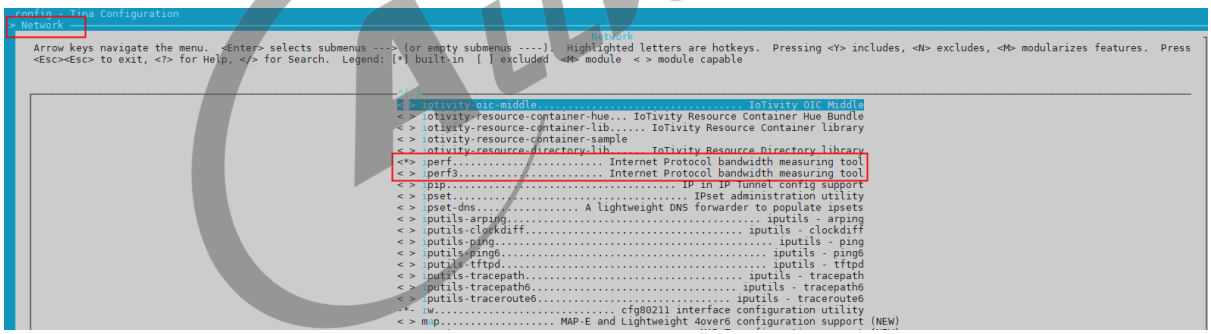


图 3-30: Tina-iperf 工具配置

6. wpa 工具配置

wpa 工具包括 wpa-supplicant 服务的和 wpa-cli 客户端。

```
make menuconfig
 > Network
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

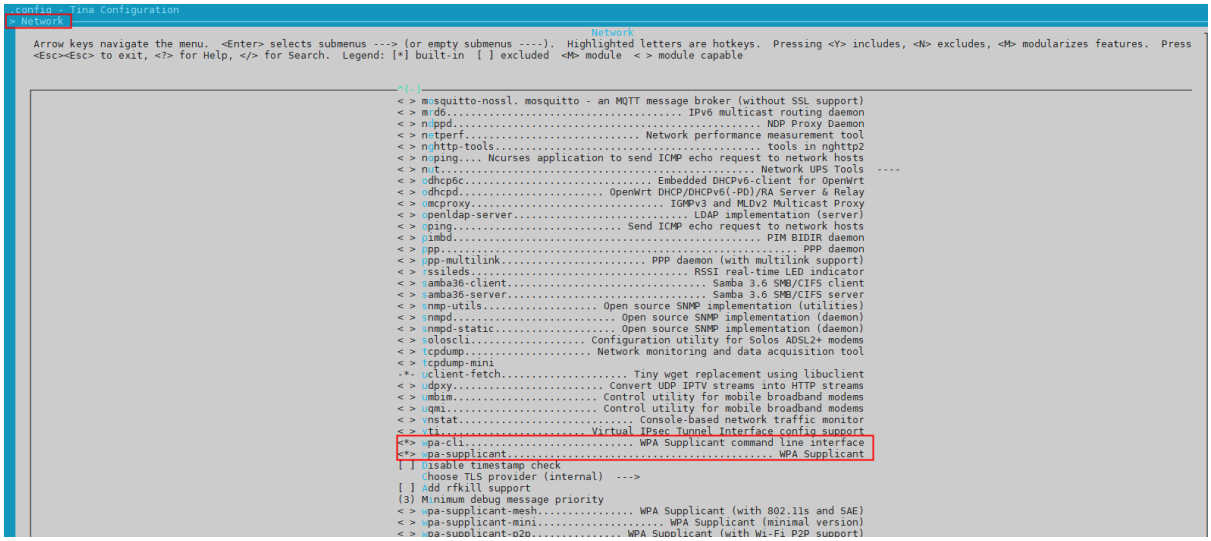


图 3-31: Tina-wpasupplicant 配置

3.2.2 RTL8821CS 模组移植

主控：R528

无线模组：RTL8821CS

内核版本：linux-5.4

方案：r528_evb1-tina

3.2.2.1 内核驱动适配

内核驱动适配分为如下几个步骤：

- 获取源码。
- 添加 Kconfig 和 Makefile 配置。
- 修改驱动源码
- make kernel_menuconfig 配置。
- 编译。

1）获取 RTL8821CS 驱动源码，放到内核驱动路径下。【请直接联系原厂获取】

```
Tina4.0/lichee/linux-5.4/drivers/net/wireless/rtl8821cs/
├──── clean
├──── core
├──── hal
├──── halmac.mk
├──── ifcfg-wlan0
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
├──── include
├──── Kconfig
├──── Makefile
├──── modules.order
├──── os_dep
├──── platform
├──── rtl8821c.mk
├──── runwpa
└──── wlan0dhcp
```

2）内核添加 Kconfig 和 Makefile 的配置。

Tina4.0/lichee/linux-5.4/drivers/net/wireless/Kconfig 文件中引入 RTL8821CS 驱动的 Kconfig 配置。

```
source "drivers/net/wireless/rtl8821cs/Kconfig"
```

Tina4.0/lichee/linux-5.4/drivers/net/wireless/Makefile 文件中引入 RTL8821CS 驱动的 Makefile 配置。

```
obj-$(CONFIG_RTL8821CS)   += rtl8821cs/
```

3）修改驱动源码

Tina4.0/lichee/linux-5.4/drivers/net/wireless/rtl8821cs/修改驱动原生代码的 Makefile。

```
CONFIG_RTW_ANDROID = 0       //0: no Android, 4/5/6/7/8/9/10 : Android version
CONFIG_PLATFORM_I386_PC = n    //原厂默认提供的是针对i386的平台适配
CONFIG_PLATFORM_ARM_SUNxI = y  //平台宏适配改为allwinner
CONFIG_RTW_DEBUG = y         //debug宏控制，适配初期建议打开
CONFIG_RTW_LOG_LEVEL = 3     //debug等级调整，0-4,建议设置成3，确定是驱动的问题再设置为4跟踪
```

Tina4.0/lichee/linux-5.4/drivers/net/wireless/rtl8821cs/platform 替换平台文件，主要涉及上下电，扫卡等函数。

```
platform_ARM_SUNxI_sdio.c
platform_ARM_SUNnI_sdio.c
```

平台文件可以从已经移植的 RTL 模组中替换到该路径，或者按照第二章说明逐步实现上电扫卡函数。

4）make kernel_menuconfig 配置

a. RTL8821CS 驱动配置

在 Tina4.0/lichee/linux-5.4 内核目录执行 m kernel_menuconfig

```
> Device Drivers > Network device support > Wireless LAN
   <M>   Realtek 8821C SDIO WiFi
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

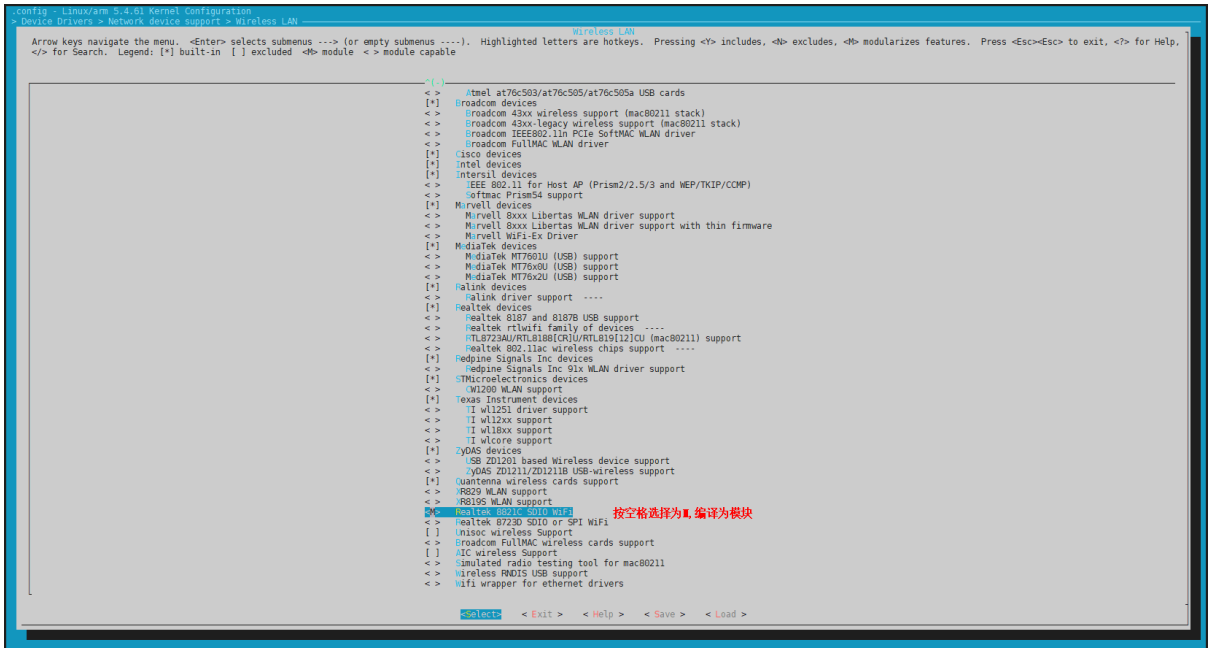


图 3-32: Tina-RTL8821CS 内核配置

b. sunxi-rf 配置

> Device Drivers > Misc devices
<*> Allwinner rfkill driver

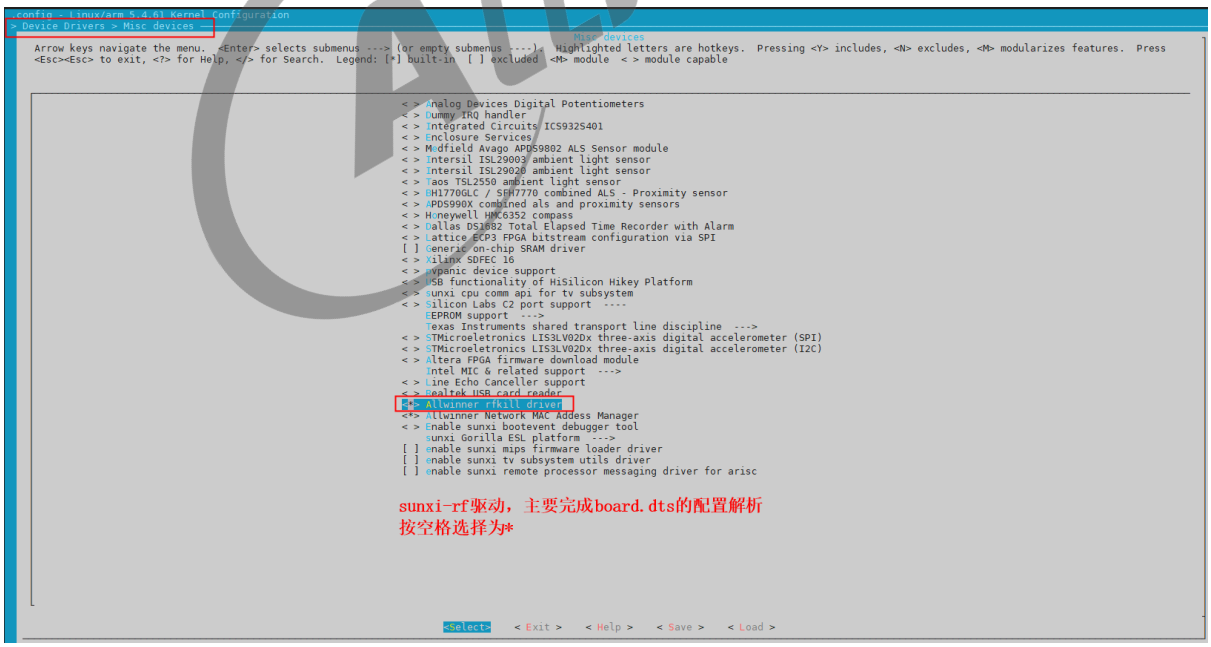


图 3-33: Tina-sunxi-rf 配置

c. SDIO 配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
> Device Drivers > MMC/SD/SDIO card support
  <*>  Allwinner sunxi SD/MMC Host Controller support  --->
```

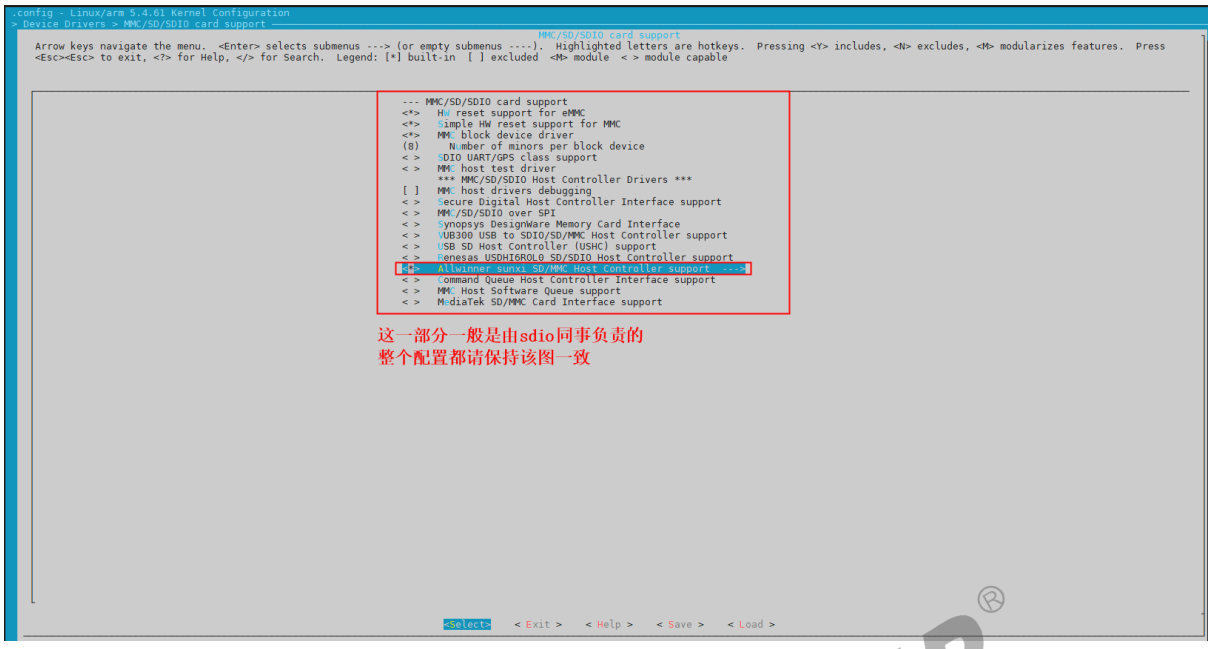


图 3-34: Tina-sdc 配置

5) 编译

在 Tina4.0/lichee/linux-5.4 内核目录执行 mkernel 进行编译。

```
Tina4.0/lichee/linux-5.4]$ mkernel
Tina4.0/lichee/linux-5.4/drivers/net/wireless/rtl8821cs/
├──── 8821cs.ko          //正常编译出8723ds.ko模块
├──── 8821cs.mod
├──── 8821cs.mod.c
├──── 8821cs.mod.o
├──── 8821cs.o
├──── clean
├──── core
├──── hal
├──── halmac.mk
├──── ifcfg-wlan0
├──── include
├──── Kconfig
├──── Makefile
├──── modules.order
├──── os_dep
├──── platform
├──── rtl8821c.mk
├──── runwpa
└──── wlan0dhcp
```

3.2.2.2 硬件资源适配

在 Tina4.0/device/config/chips/r528/configs/evb1/linux/board.dts 中添加引脚配置。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
&pio {
wlan_pins_a:wlan@0 {
        pins = "PG11";
        function = "clk_fanout1";  //这里涉及一个特殊处理，该方案的32k采用PG11复用给出
    };
...
};

&soc {
    rfkill: rfkill@0 {
        compatible   = "allwinner,sunxi-rfkill";
        chip_en;
        power_en;
        pinctrl-0 = <&wlan_pins_a>;
        pinctrl-names = "default";
        status      = "okay";

        wlan: wlan@0 {
            compatible   = "allwinner,sunxi-wlan";
            clock-names = "32k-fanout1";
            clocks = <&ccu CLK_FANOUT1_OUT>;
            wlan_busnum   = <0x1>;
            wlan_regon   = <&pio PE 17 GPIO_ACTIVE_HIGH>;
            wlan_hostwake  = <&pio PG 10 GPIO_ACTIVE_HIGH>;
            /*wlan_power   = "VCC-3V3";*/
            /*wlan_power_vol = <3300000>;*/
            /*interrupt-parent = <&pio>;
            interrupts = < PG 10 IRQ_TYPE_LEVEL_HIGH>;*/
            wakeup-source;

        };

...
    };
};
```

属性	说明
clocks	用于配置使用主控提供的 32k 时钟
pinctrl-0	用于配置 pin 的复用功能
pinctrl-names	用于配置 pin state
wlan_busnum	表示 WiFi 所使用的 SDIO 控制器号
wlan_power	表示给 WiFi 模组供电的 regulator 名称
wlan_io_regulator	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_regon	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_hostwake	表示 WiFi 唤醒主控的 GPIO
chip_en	表示 WiFi 模组使能引脚，硬件未使用时不配置
power_en	表示模块外部的电源开关控制引脚

以上所有项必须参看原理图进行配置，配置与原理图实际使用的资源保持一致；最好是和硬件同事一起确认，比如有些设计供电可能是没有 axp 的，硬件直接供电了，所以不需要配置，特别注意的就是 sdio 的配置。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

说明

特别注意，特别注意，特别注意因为 rtl8821cs 的模组是 sdio3.0 的，所以 VCCIO_WIFI 需要适配为 1.8V，才能发挥出最优性能，当然 3.3V 也可以工作起来。

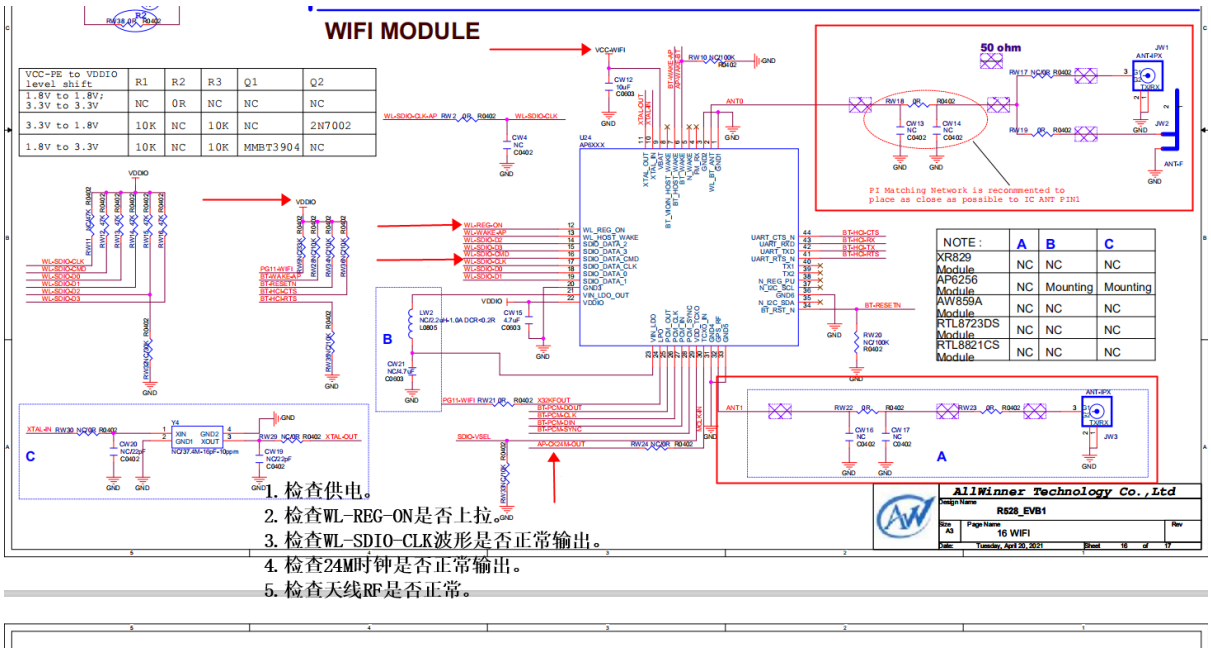
例如：linux-4.9 内核中：Tina4.0/device/config/chips/r818/configs/scanp_rtl8821cs/board.dts

```
sdc1: sdmmc@04021000 {
                bus-width = <4>;
                no-mmc;
                no-sd;
                cap-sd-highspeed;
                /*sd-uhs-sdr12*/
                /*sd-uhs-sdr25*/
                /*sd-uhs-sdr50;*/
                /*sd-uhs-ddr50;*/
                /*sd-uhs-sdr104*/
                /*sunxi-power-save-mode;*/
                /*sunxi-dis-signal-vol-sw;*/
                sdio-used-1v8;    //改用1.8V供电
                cap-sdio-irq;
                keep-power-in-suspend;
                ignore-pm-notify;
                max-frequency = <50000000>;
                ctl-spec-caps = <0x8>;
                status = "okay";
        };
```

linux-5.4 内核中：Tina4.0/device/config/chips/r528/configs/evb2/linux/evb2/board.dts。

```
&pio {
    vcc-pg-supply = <&reg_pio1_8>;  //改用1.8V供电(linux-5.4的配置涉及具体IO的调整)
    sdc0_pins_a: sdc0@0 {
        allwinner,pins = "PF0", "PF1", "PF2",
                "PF3", "PF4", "PF5";
        allwinner,function = "sdc0";
        allwinner,muxsel = <2>;
        allwinner,drive = <3>;
        allwinner,pull = <1>;
        pins = "PF0", "PF1", "PF2",
            "PF3", "PF4", "PF5";
        function = "sdc0";
        drive-strength = <30>;
        bias-pull-up;
        power-source = <3300>;
    };
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利



1. 检查供电。
2. 检查WL-REG-ON是否上拉。
3. 检查WL-SDIO-CLK波形是否正常输出。
4. 检查24M时钟是否正常输出。
5. 检查天线RF是否正常。

图 3-35: Tina-RTL8821CS 模组原理图

3.2.2.3 方案 module 适配

在 Tina4.0/target/allwinner/r528-common/modules.mk 中添加模块配置。

```
define KernelPackage/net-rtl8821cs
  SUBMENU:=$(WIRELESS_MENU)
  TITLE:=RTL8821CS support (staging)
  DEPENDS:= +rtl8821cs-firmware +@IPV6
  FILES:=$(LINUX_DIR)/drivers/net/wireless/rtl8821cs/8821cs.ko
  AUTOLOAD:=$(call AutoProbe,8821cs)
endef

define KernelPackage/net-rtl8821cs/description
 Kernel modules for RealTek RTL8821CS support
endef

$(eval $(call KernelPackage,net-rtl8821cs))
```

接着在 Tina4.0 根目录执行 m menuconfig 就可以看到新添加的模组。

```
> Kernel modules > Wireless Drivers
    <*> kmod-net-rtl8821cs..... RTL8821CS support (staging)
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

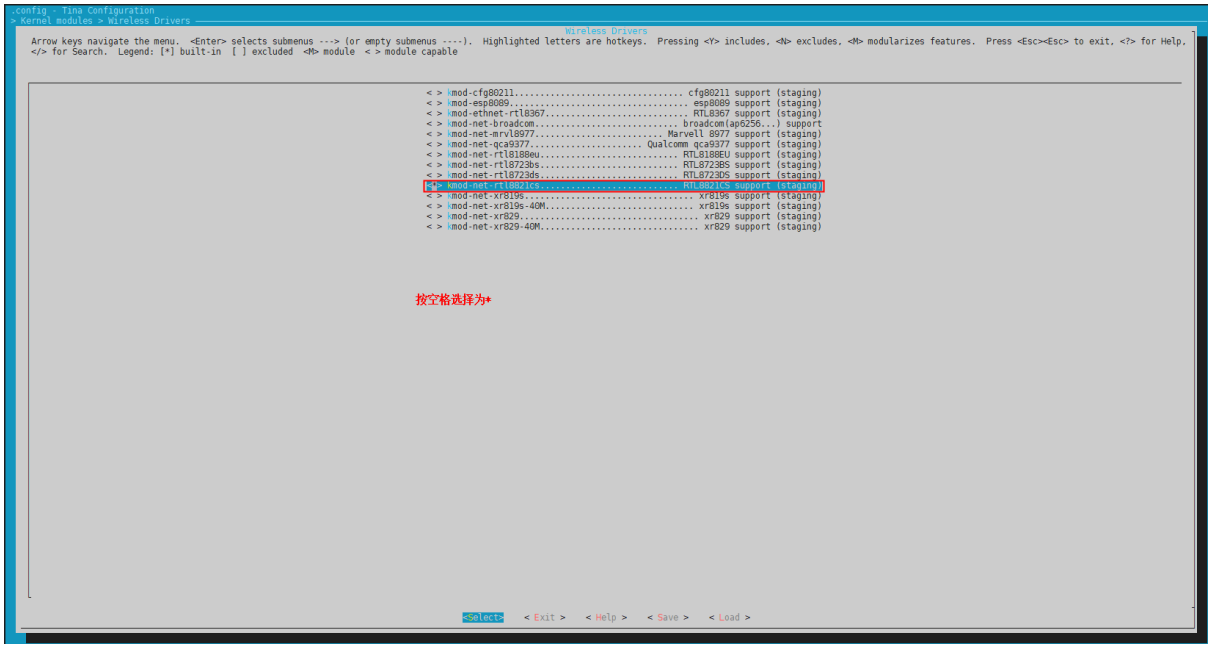


图 3-36: Tina-RTL8821CS-module 配置

3.2.2.4 添加 Firmware

在 Tina4.0/package/firmware/linux-firmware/添加 rtl8821cs 需要的 firmware。

```
Tina4.0/package/firmware/linux-firmware/rtl8821cs/
├──── rtl8821c_config
├──── rtl8821c_fw
└──── rtl8821cs.mk
```

rtl8723ds.mk 文件如下：

```
Package/rtl8821cs-firmware = $(call Package/firmware-default,RealTek RTL8821CS firmware)

define Package/rtl8821cs-firmware/install
    $(INSTALL_DIR) $(1)/$(FIRMWARE_PATH)/rtlbt
    $(INSTALL_DATA) \
        $(TOPDIR)/package/firmware/linux-firmware/rtl8821cs/rtl8821c_fw \
        $(1)/$(FIRMWARE_PATH)/rtlbt/rtl8821c_fw
    $(INSTALL_DATA) \
        $(TOPDIR)/package/firmware/linux-firmware/rtl8821cs/rtl8821c_config \
        $(1)/$(FIRMWARE_PATH)/rtlbt/rtl8821c_config
endef

$(eval $(call BuildPackage,rtl8821cs-firmware))
```

说明：可以看到整个 Makefile 文件就只是做了简单的拷贝动作。

说明

一般 firmware 的路径是：系统的/lib/firmware/, 如果更改请确保是否和驱动中定义的保持一致。

驱动中定义 firmware 文件为：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
rtl8821cs/Makefile
CONFIG_LOAD_PHY_PARA_FROM_FILE = y
...
ifeq ($(CONFIG_LOAD_PHY_PARA_FROM_FILE), y)
EXTRA_CFLAGS += -DCONFIG_LOAD_PHY_PARA_FROM_FILE
#EXTRA_CFLAGS += -DREALTEK_CONFIG_PATH_WITH_IC_NAME_FOLDER
EXTRA_CFLAGS += -DREALTEK_CONFIG_PATH=\"/lib/firmware/\"
endif
```

接着在 Tina4.0 根目录执行 m menuconfig 就可以看到新添加的 firmware 配置。

```
> Firmware
    <*> rtl8821cs-firmware..... RealTek RTL8821CS firmware
```

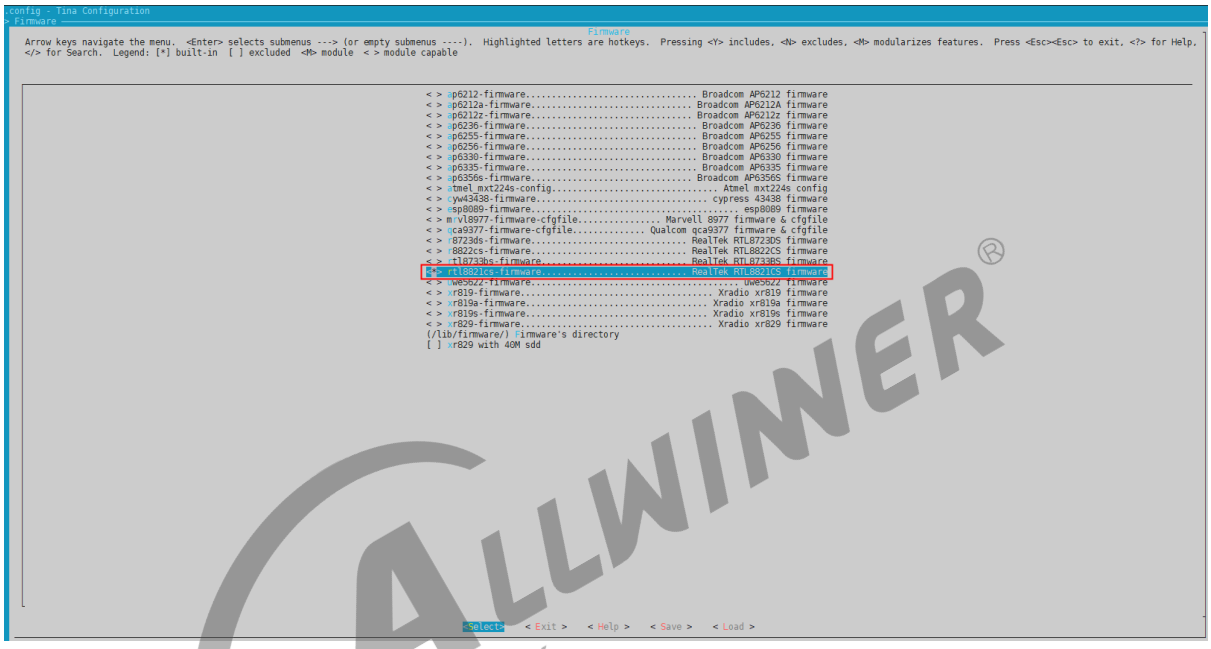


图 3-37: Tina-RTL8821CS-firmware 配置

3.2.2.5 应用工具适配

1. wifimanager 配置

```
make menuconfig
> Allwinner > wifimanager
```

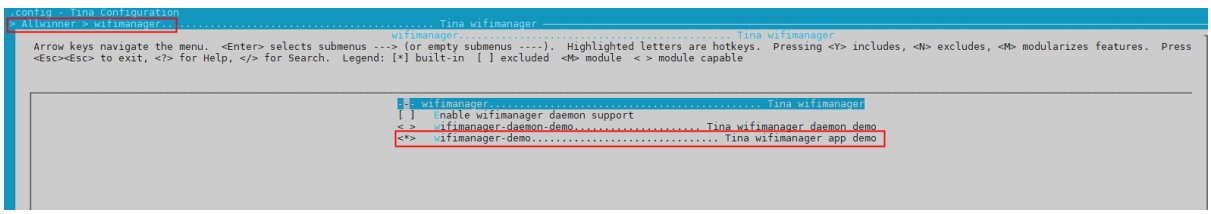


图 3-38: Tina-wifimanager 配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

2. smartlinkd 配网配置

make menuconfig
> Allwinner > smartlinkd

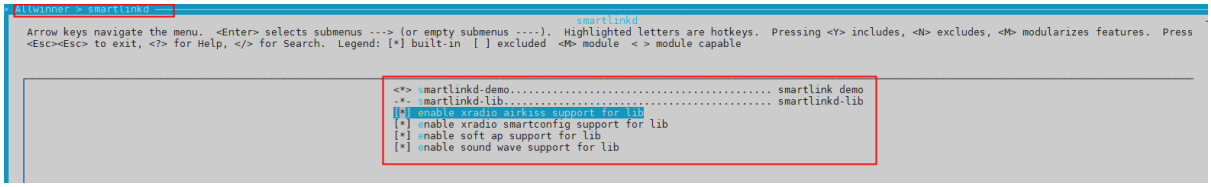


图 3-39: Tina-smartlinkd 配置

3. softap 配置

make menuconfig
> Allwinner > softap

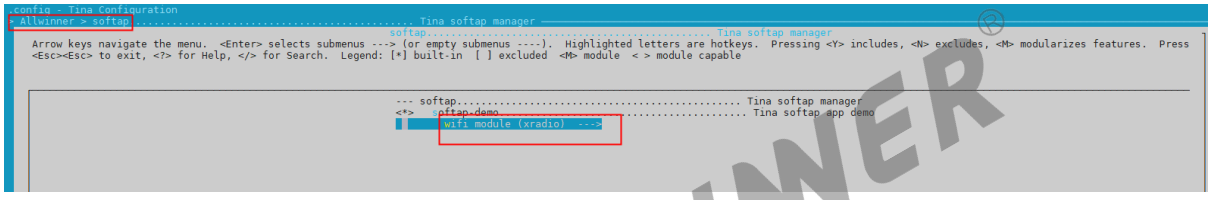


图 3-40: Tina-softap 配置

4. rf 工具配置

make menuconfig
> Utilities > rf test tool

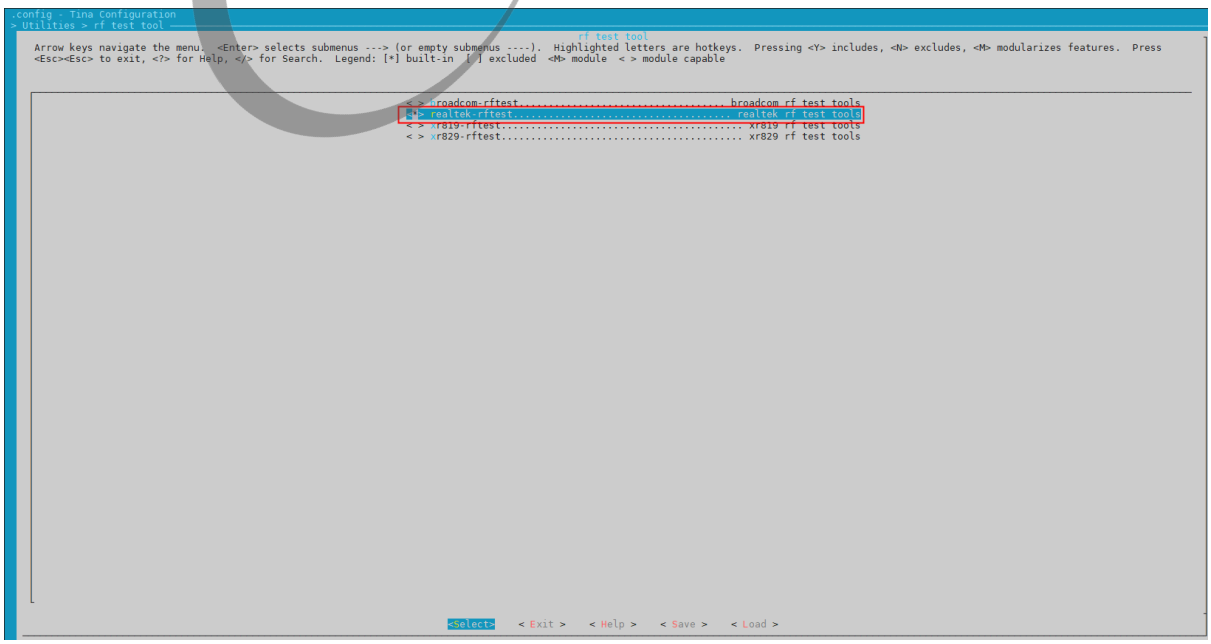


图 3-41: Tina-RTL-rf 工具配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5. iperf 工具配置

```
make menuconfig
> Network
```

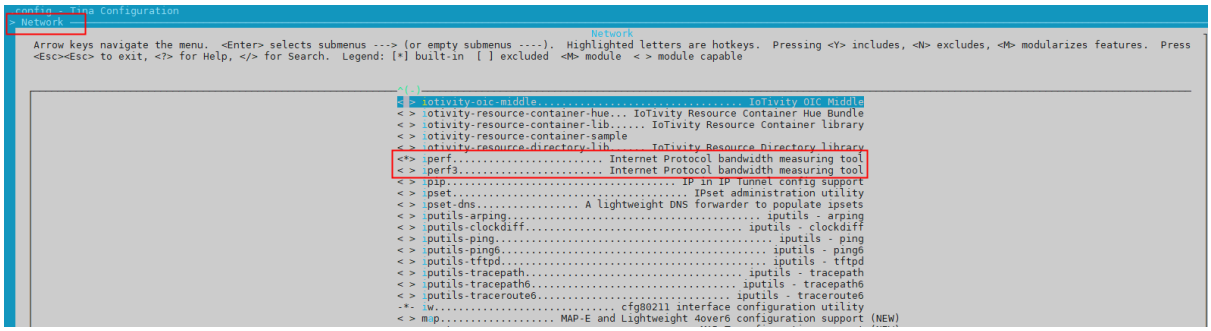


图 3-42: Tina-iperf 工具配置

6. wpa 工具配置

wpa 工具包括 wpa-supplicant 服务的和 wpa-cli 客户端

```
make menuconfig
 > Network
```

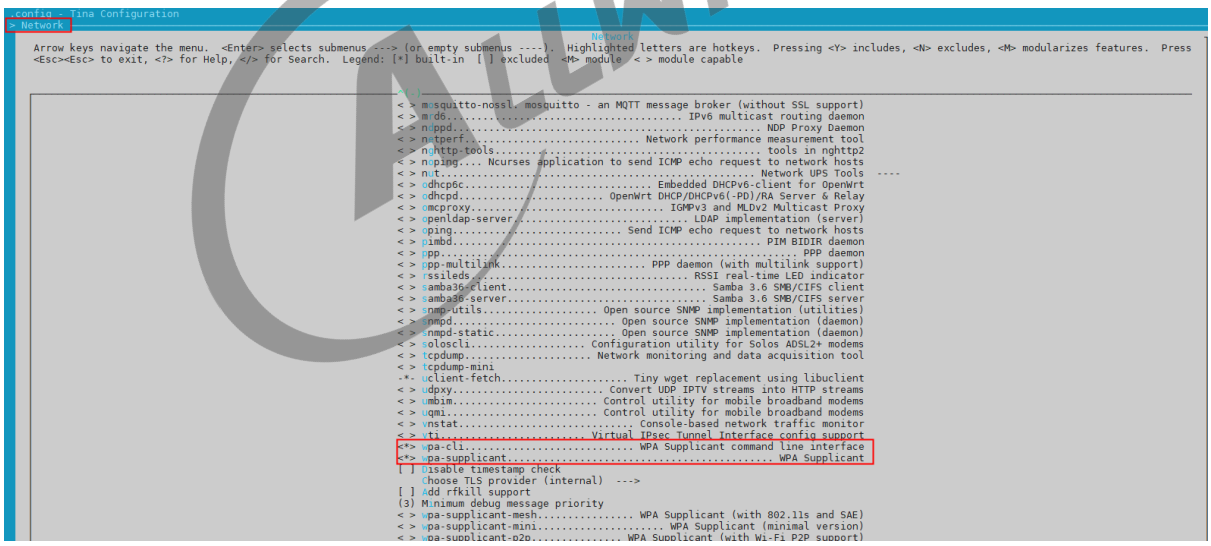


图 3-43: Tina-wpasupplicant 配置

3.3 乐鑫 ESP 系列

乐鑫的无线模组型号多种多样，目前我司搭配的方案主要有:

版权所有 © 珠海全志科技股份有限公司。保留一切权利

表 3-7: ESP 支持列表

Vendor	PartNumber	Interface	Wi-Fi	BT
ESPRESSIF	esp32	SDIO	b/g/n	支持

3.3.1 ESP32 模组移植

主控：MR133

无线模组：乐鑫 esp32

内核版本：linux-4.9

方案：mr133_robots-tina

说明

乐鑫该款 esp32 模组不是标准的 wifi 模组，它实质上是一款 iot 的 mcu 芯片，与通常的 wifi 模组有本质上的差别，在启动和配置上都与通常的 wifi 模组有差异。

3.3.1.1 内核驱动适配

内核驱动适配分为如下几个步骤：

- 获取源码。
- 添加 common 文件。
- 移动头文件到 include 目录统一进行管理。
- 添加平台文件。
- 添加 Kconfig 和 Makefile 配置。
- 修改驱动源码。
- make kernel_menuconfig 配置。
- 编译。

1）获取 esp32 驱动源码，放到内核驱动路径下。客户可以从该链接上获取。

https://github.com/espressif/esp-hosted ，一般随 SDK 直接适配释放。【请直接联系原厂】

该网址下载到的驱动源码是 esp32 的开发包，对于 linux 平台来说只用到其中的部分文件。

linux 驱动源码位置在 esp-hosted/host/linux/host_driver/esp32，把源码移植到 tina 上。

```
Tina4.0/lichee/linux-4.9/drivers/net/wireless/esp32/
├── sdio
├── spi
├── LICENSE
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
├── Makefile
├── esp.h
├── esp_api.h
├── esp_bt.c
├── esp_bt_api.h
├── esp_if.h
├── esp_rb.c
├── esp_rb.h
├── esp_serial.c
├── esp_serial.h
└── main.c
```

2) 添加 common 里的头文件用于解决编译问题，代码路径为 esp-hosted/common/include/。

common 里的头文件 stm32 以及 Linux 平台都需要使用，因此需要移植到驱动中。

```
Tina4.0/lichee/linux-4.9/drivers/net/wireless/esp32/
├── common
│   └── include
│       ├── adapter.h
│       └── esp_hosted_config.pb-c.h
├── sdio
├── spi
├── LICENSE
├── Makefile
├── esp.h
├── esp_api.h
├── esp_bt.c
├── esp_bt_api.h
├── esp_if.h
├── esp_rb.c
├── esp_rb.h
├── esp_serial.c
├── esp_serial.h
└── main.c
```

3) 添加 include 目录用于解决编译时头文件寻找问题。把驱动中所有头文件移到 include 目录里。

内核编译时为了方便处理头文件连接问题统一把头文件放到 include 目录。

```
Tina4.0/lichee/linux-4.9/drivers/net/wireless/esp32/
├── common
├── sdio
├── spi
├── LICENSE
├── Makefile
├── include
│   ├── esp_api.h
│   ├── esp_bt_api.h
│   ├── esp.h
│   ├── esp_if.h
│   ├── esp_rb.h
│   ├── esp_sdio_api.h
│   ├── esp_sdio_decl.h
│   └── esp_serial.h
├── esp_bt.c
├── esp_rb.c
├── esp_serial.c
└── main.c
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

4) 添加 Tina4.0/lichee/linux-4.9/drivers/net/wireless/esp32/platform 平台文件，主要涉及上下电，扫卡等函数。

平台文件可以从已经移植的 wifi 模组中替换到该路径，如果你拿到的 SDK 都没有适配过任意一款 esp 模组，请联系我们。

```
Tina4.0/lichee/linux-4.9/drivers/net/wireless/esp32/
├──── common
├──── sdio
├──── spi
├──── LICENSE
├──── Makefile
├──── include
│   └──── platform_ops.h
├──── platform
│   └──── platform_ARM_SUNxI_sdio.c
├──── esp_bt.c
├──── esp_rb.c
├──── esp_serial.c
└──── main.c
```

5) 内核添加 Kconfig 和 Makefile 的配置。

Tina4.0/lichee/linux-4.9/drivers/net/wireless/Kconfig 文件中引入 esp32 驱动的 Kconfig 配置。

```
source "drivers/net/wireless/esp32/Kconfig"
```

Tina4.0/lichee/linux-4.9/drivers/net/wireless/Makefile 文件中引入 esp32 驱动的 Makefile 配置。

```
obj-$(CONFIG_ESP32_WLAN)  += esp32/
```

文件添加完成后源码目录结构变化如下：

```
Tina4.0/lichee/linux-4.9/drivers/net/wireless/esp32/
├──── common
├──── sdio
├──── spi
├──── LICENSE
├──── Makefile
├──── include
├──── platform
├──── esp_bt.c
├──── esp_rb.c
├──── esp_serial.c
└──── main.c
```

6) 修改驱动源码

Tina4.0/lichee/linux-4.9/drivers/net/wireless/esp32/修改驱动原生代码的 Makefile。

添加头文件寻找路径

```
NOSTDINC_FLAGS:= -I$(srctree)/drivers/net/wireless/esp32/common/include/
NOSTDINC_FLAGS+= -I$(srctree)/drivers/net/wireless/esp32/include/
```

add wifi power module //添加驱动上下电文件

版权所有 © 珠海全志科技股份有限公司。保留一切权利

module_objects += platform/platform_ARM_SUNXI_sdio.o

在驱动加载流程中添加上下电，扫卡函数，用于加载时启动上下电以及扫卡动作。

在 main.c 文件的驱动加载卸载函数中添加上下电扫卡接口。

```
static int __init esp_init(void)
{
   ......
   /*power on*/
   ret = platform_wifi_power_on();
   ......
}

static void __exit esp_exit(void)
{
   ......
   platform_wifi_power_off();
   ......
}
```

修改驱动 tx 接口适配 XRADIOsdio 接口。

esp32 驱动存在数据地址没有 4 字节对齐的问题，该问题会导致部分 wifi，bt 命令无法通过 sido 口发送到 esp32 模块，需要修改 wifi 和 bt 的 tx 接口函数适配 XRADIOsdio 接口。该问题补丁修改比较多，不在此列举，可以参考 tina 4.9 内核提交 32dbac2ea5 和 0ba8c00071。

7）make kernel_menuconfig 配置

在 Tina4.0/lichee/linux-4.9 内核目录执行 m kernel_menuconfig

```
> Device Drivers > Network device support > Wireless LAN
   <M>   ESP32 WLAN support
```

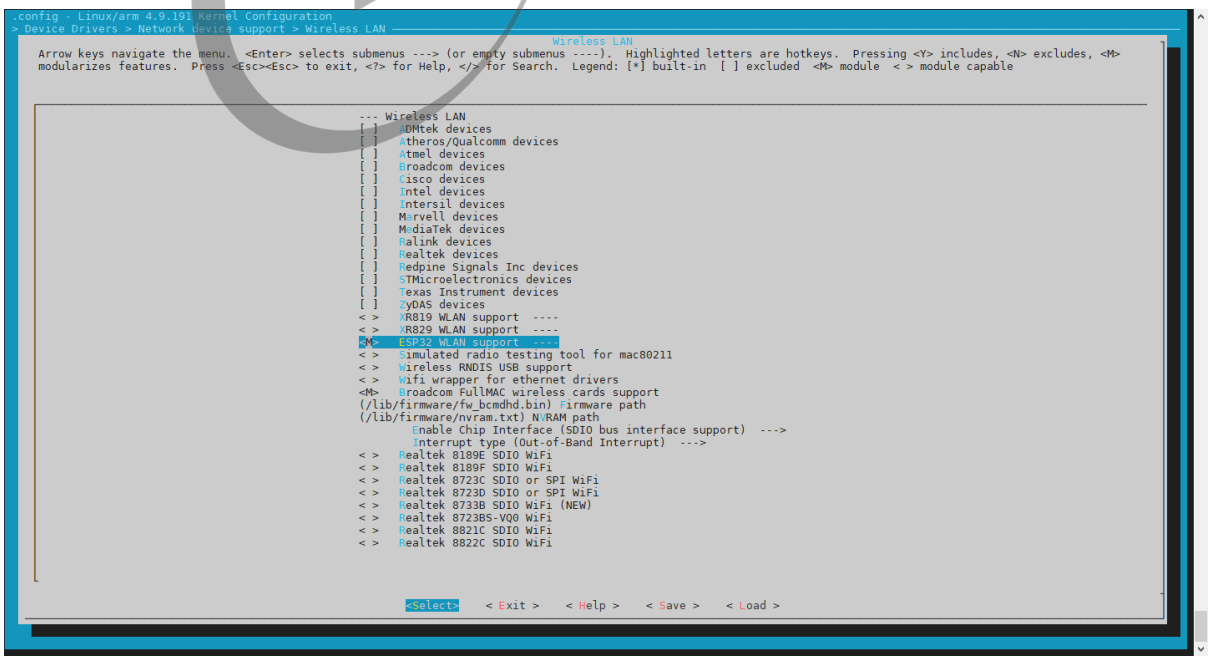


图 3-44: Tina-esp32 内核配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

8）编译

在 Tina4.0/lichee/linux-4.9 内核目录执行 mkernel 进行编译。

Tina4.0/lichee/linux-4.9]$ mkernel

```
Tina4.0/lichee/linux-4.9/drivers/net/wireless/esp32/
├──── common
├──── esp32_sdio.ko      //正常编译出来的esp32驱动
├──── esp32_sdio.mod.o
├──── esp32_sdio.mod.c
├──── esp_bt.c
├──── esp_rb.c
├──── esp_serial.c
├──── include
├──── Kconfig
├──── LICENSE
├──── main.c
├──── Makefile
├──── platform
├──── sdio
└──── spi
```

3.3.1.2 硬件资源适配

3.3.1.3 方案 module 适配

在 Tina4.0/target/allwinner/mr133-common/modules.mk 中添加模块配置。

```
define KernelPackage/net-esp32
 SUBMENU:=$(WIRELESS_MENU)
 TITLE:=ESP32 support (staging)
 DEPENDS:= +@IPV6
 FILES:=$(LINUX_DIR)/drivers/net/wireless/esp32/esp32_sdio.ko
 AUTOLOAD:=$(call AutoProbe,esp32)
endef

define KernelPackage/net-esp32/description
 Kernel modules for esp32 support
endef

$(eval $(call KernelPackage,net-esp32))
```

3.3.1.4 添加 Firmware

因为 esp32 是特殊的 mcu 模组，在模组加载时没有下载 firmware 的动作，firmware 实质上是以 mcu 系统的方式已经烧写到 esp32 芯片里了，若要更换 firmware 需要联系乐鑫进行 mcu 系统固件的烧写。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3.3.1.5 应用工具适配

1. esp32 工具配置

其他平台如果支持 paytion 可以使用 paytion 版本的配置工具，paytion 版本配置工具可以从 esp32 开发包 esp-hosted/host/linux/host_control/python_support 目录获取。c 版本配置工具源码在 esp32 开发包 esp-hosted/host/linux/host_control/c_support，tina 上只提供 c 语言版本的配置工具，c 语言版本进行下面配置选择后可进行编译。

```
make menuconfig
> Allwinner > esp_c_support
```

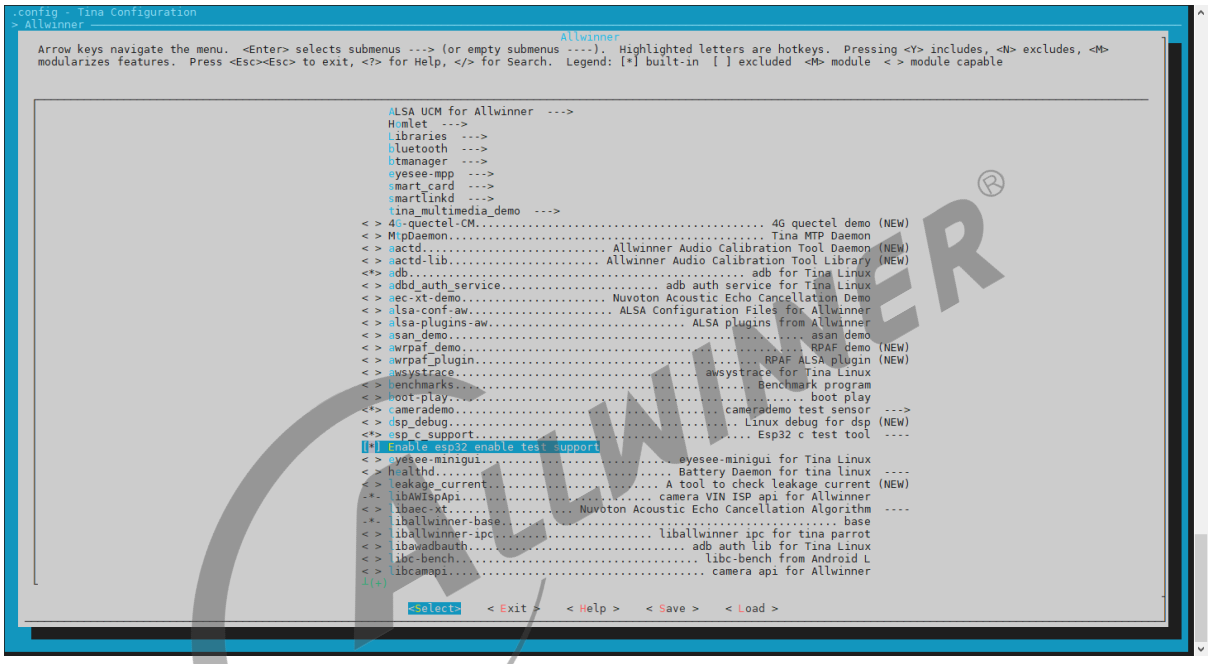


图 3-45: Tina-esp32 工具配置

需要注意的是 python 版本的配置工具已进行分割，即 ap_scan_list,connect_stations_list,softap_config, 等等命令都是独立一个脚本或命令的。但 c 版本没有进行切分，所有命令都集成在了一个 c 文件中，该文件编译出来的应用只适合测试用，真正开发时。需要用户对该 c 文件进行切分，切分成类似 paython 工具那样的单独命令。

2. iperf 工具配置

```
make menuconfig
> Network
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

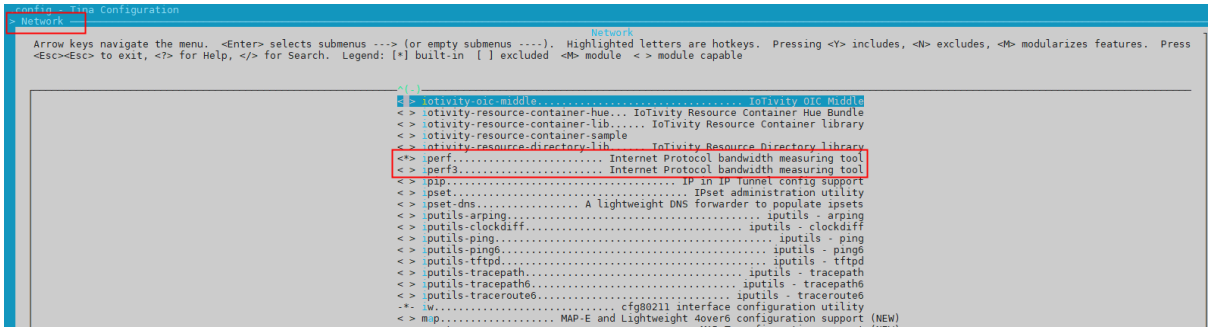


图 3-46: Tina-iperf 工具配置

3. RF 测试工具的适配

因为 esp32 是特殊的 mcu 模组，没有 RF 测试工具，如需要 RF 测试数据请直接联系乐鑫。

3.4 AIC 系列

3.4.1 AIC8800 模组移植

主控：D1

无线模组：AW869B

内核版本：linux-5.4

方案：D1_neza-Tina

说明

该款 Wifi 模组 IO 电压要求 1.8V

3.4.1.1 内核驱动适配

内核驱动适配分为如下四个步骤：

- 获取源码。
- 添加 Kconfig 和 Makefile 配置。
- make kernel_menuconfig 配置。
- 编译。

1）获取驱动源码，放到内核驱动路径下。客户可以从一号通账号获取。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
Tina4.0/lichee/linux-5.4/drivers/net/wireless/aic8800
├──── aic8800_bsp
├──── aic8800_btlpm
├──── aic8800_fdrv
├──── Kconfig
└──── Makefile
```

2）内核添加 Kconfig 和 Makefile 的配置。

Tina4.0/lichee/linux-5.4/drivers/net/wireless/Kconfig 文件中引入 AW869B 驱动的 Kconfig 配置。

```
source "drivers/net/wireless/aic8800/Kconfig"
```

Tina4.0/lichee/linux-5.4/drivers/net/wireless/Makefile 文件中引入 AW869B 驱动的 Makefile 配置。

```
obj-$(CONFIG_AIC_WLAN_SUPPORT)   += aic8800/
```

3）make kernel_menuconfig 配置

在 Tina4.0/lichee/linux-5.4 内核目录执行 m kernel_menuconfig。

a. aic8800 驱动的配置

```
> Device Drivers > Network device support > Wireless LAN
    <*> AIC wireless Support
    <M>  AIC8800 wlan Support
    <M>  AIC8800 bluetooth Support
```

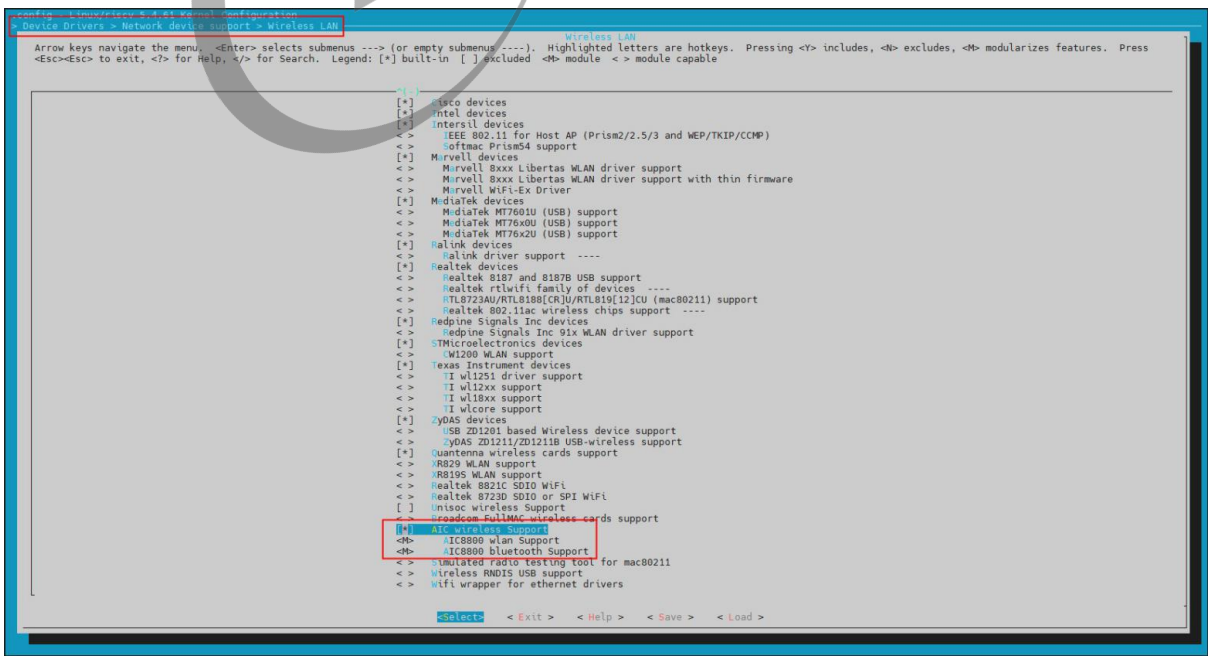


图 3-47: Tina-aic8800 内核配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

b. sunxi-rf 的配置

```
> Device Drivers > Misc devices
    <*> Allwinner rfkill driver
```

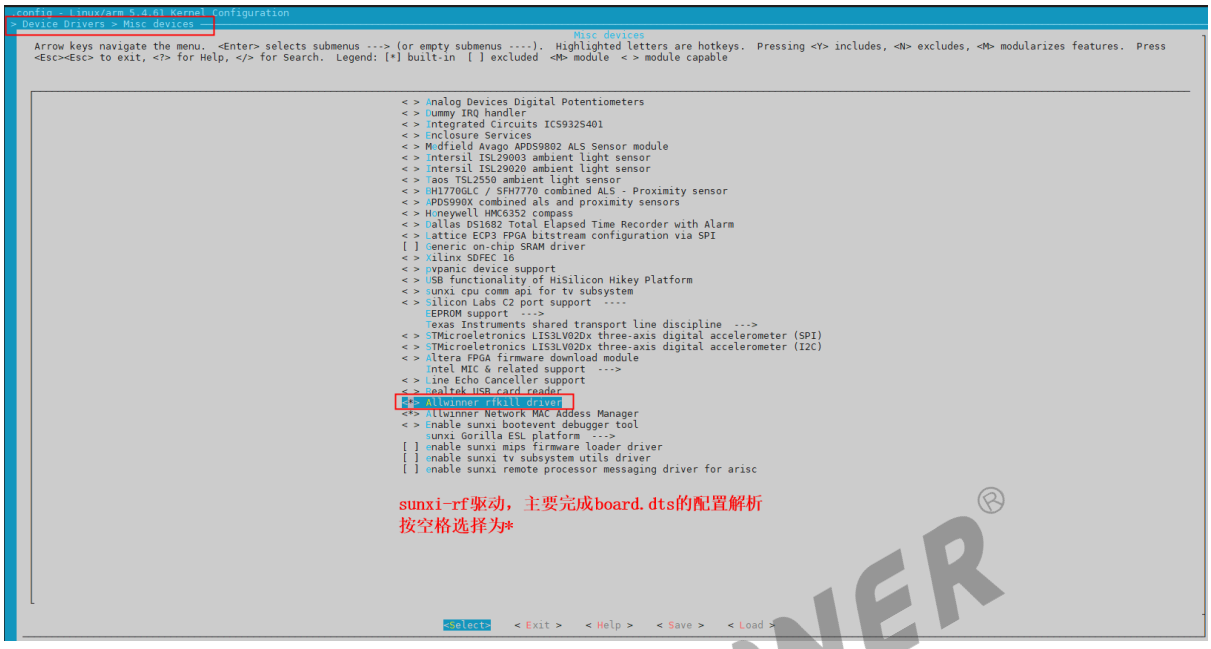


图 3-48: Tina-sunxi-rf 配置

c. SDIO 的配置

```
> Device Drivers > MMC/SD/SDIO card support
    <*> Allwinner sunxi SD/MMC Host Controller support --->
```

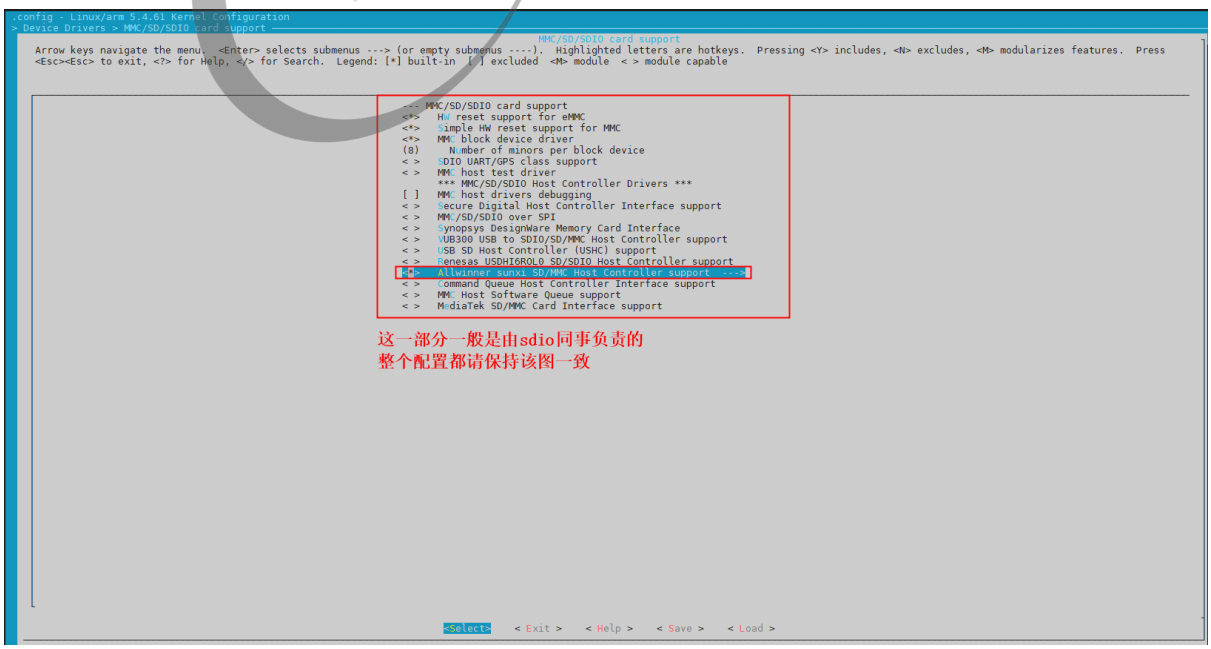


图 3-49: Tina-sdc 配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

4）编译

在 Tina4.0/lichee/linux-5.4 内核目录执行 mkernel 进行编译。

```
├──── aic8800_bsp
│    ├──── aic8800_bsp.ko
│    ├──── ...
├──── aic8800_btlpm
│    ├──── aic8800_btlpm.ko
│    ├──── ...
├──── aic8800_fdrv
│    ├──── aic8800_fdrv.ko
│    ├──── ...
├──── built-in.a
├──── Kconfig
├──── Makefile
├──── modules.builtin
└──── modules.order
```

3.4.1.2 硬件资源适配

在 Tina4.0/device/config/chips/d1/configs/nezha/linux-5.4/board.dts 中添加引脚配置。

```
&pio {
    vcc-pg-supply = <&reg_pio1_8>; /*AIC8800的IO电压为1.8V，这里需要配置PG口1.8V*/
    sdc0_pins_a: sdc0@0 {
        ...
    };
...
    wlan_pins_a:wlan@0 {
        pins = "PG11";
        function = "clk_fanout1";
    };
};

&soc {
    ...
    rfkill: rfkill@0 {
        compatible    = "allwinner,sunxi-rfkill";
        chip_en;
        power_en;
        pinctrl-0 = <&wlan_pins_a>;
        pinctrl-names = "default";
        status      = "okay";

        wlan: wlan@0 {
            compatible    = "allwinner,sunxi-wlan";
            clock-names = "32k-fanout1";
            clocks = <&ccu CLK_FANOUT1_OUT>;
            wlan_busnum    = <0x1>;
            wlan_regon    = <&pio PG 12 GPIO_ACTIVE_HIGH>;
            wlan_hostwake  = <&pio PG 10 GPIO_ACTIVE_HIGH>;
            /*wlan_power    = "VCC-3V3";*/
            /*wlan_power_vol = <3300000>;*/
            /*interrupt-parent = <&pio>;
            interrupts = < PG 10 IRQ_TYPE_LEVEL_HIGH>;*/
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
            wakeup-source;

        };
        ...
    };
};
```

属性	说明
clocks	用于配置使用主控提供的 32k 时钟
pinctrl-0	用于配置 pin 的复用功能
pinctrl-names	用于配置 pin state
wlan_busnum	表示 WiFi 所使用的 SDIO 控制器号
wlan_power	表示给 WiFi 模组供电的 regulator 名称
wlan_io_regulator	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_regon	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_hostwake	表示 WiFi 唤醒主控的 GPIO
chip_en	表示 WiFi 模组使能引脚，硬件未使用时不配置
power_en	表示模块外部的电源开关控制引脚

说明

以上所有项必须参看原理图进行配置，配置与原理图实际使用的资源保持一致；最好是和硬件同事一起确认，比如有些设计供电可能是没有 axp 的，硬件直接供电了，所以不需要配置，特别注意的就是 sdio 的配置。

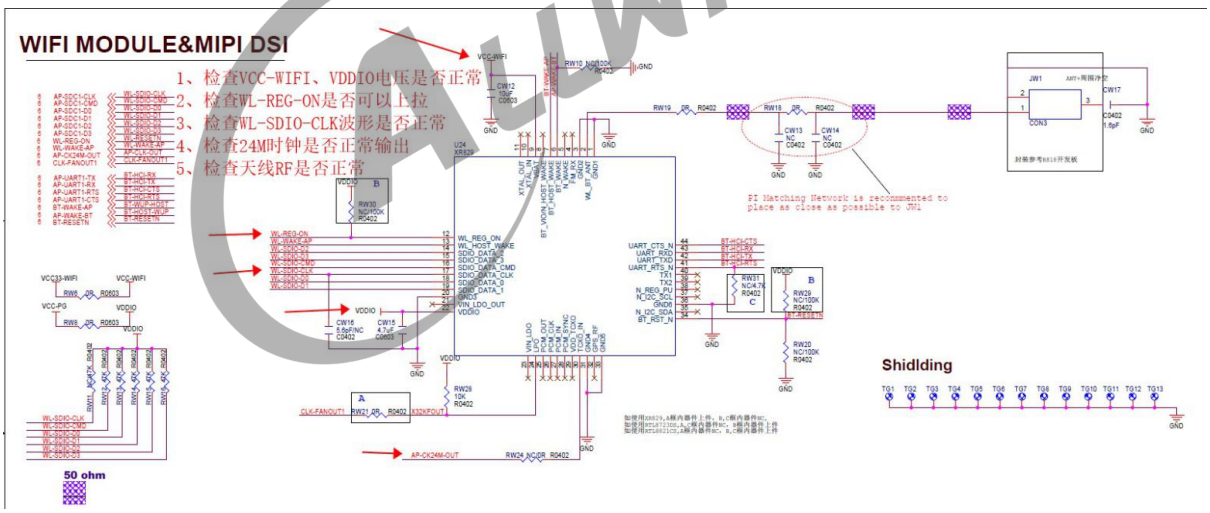


图 3-50: AW869B 与主控连接原理图

3.4.1.3 方案 module 适配

在 target/allwinner/d1-common/modules.mk 添加模块配置。

```
define KernelPackage/net-aic8800
  SUBMENU:=$(WIRELESS_MENU)
  TITLE:=aic8800 support (staging)
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
  DEPENDS:= +@IPV6
  KCONFIG:=\
    CONFIG_AIC8800_BTLPM_SUPPORT=m \
    CONFIG_AIC8800_WLAN_SUPPORT=m \
    CONFIG_AIC_WLAN_SUPPORT=m \
    CONFIG_PM=y\
    CONFIG_RFKILL=y \
    CONFIG_RFKILL_PM=y \
    CONFIG_RFKILL_GPIO=y

  FILES+=$(LINUX_DIR)/drivers/net/wireless/aic8800/aic8800_bsp/aic8800_bsp.ko
  FILES+=$(LINUX_DIR)/drivers/net/wireless/aic8800/aic8800_btlpm/aic8800_btlpm.ko
  FILES+=$(LINUX_DIR)/drivers/net/wireless/aic8800/aic8800_fdrv/aic8800_fdrv.ko

  AUTOLOAD:=$(call AutoProbe, aic8800_bsp aic8800_btlpm aic8800_fdrv)
endef

define KernelPackage/net-aic8800/description
 Kernel modules for aic8800 support
endef

$(eval $(call KernelPackage,net-aic8800))
```

几点说明：

- DEPENDS：Tina 包依赖配置
- KCONFIG：内核依赖配置
- FILES：内核模块路径

一般不需要用户自己适配，可以自己从已经适配过的方案的 modules.mk 中拷贝过来就好，也可以直接拷贝这段代码。

接着在 Tina4.0 根目录执行 m menuconfig 就可以看到新添加的模组。

```
source build/envsetup.sh
lunch d1_nezha-tina  //根据实际方案选定

make menuconfig
> Kernel modules > Wireless Drivers
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

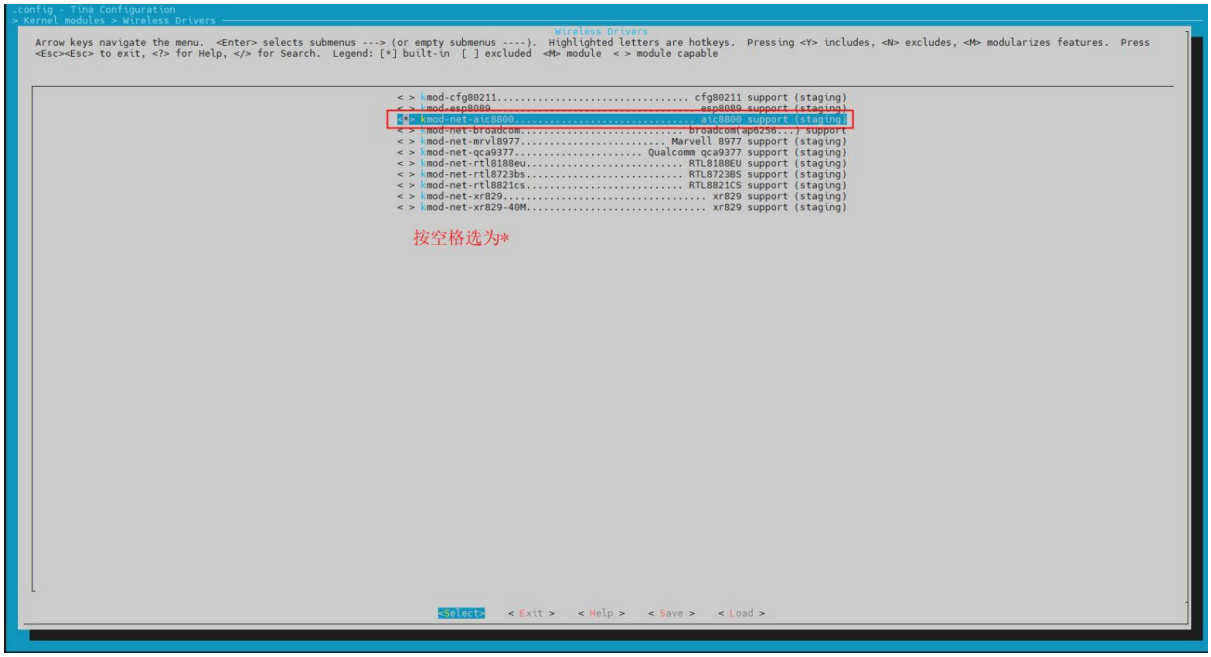


图 3-51: Tina-aic8800-module 配置

3.4.1.4 添加 Firmware

在 Tina4.0/package/firmware/linux-firmware/添加 aic8800 需要的 firmware。

```
Tina4.0/package/firmware/linux-firmware/aic8800
 ├──── aic8800.mk
 ├──── fmacfw.bin
 ├──── fmacfw_rf.bin
 ├──── fw_adid.bin
 ├──── fw_patch.bin
 ├──── fw_patch_bt_only.bin
 ├──── fw_patch_combo.bin
 ├──── fw_patch_table.bin
 └──── fw_patch_test.bin
```

配置 aic8800.mk 文件

```
Package/aic8800-firmware = $(call Package/firmware-default,AIC aic8800 firmware)

define Package/aic8800-firmware/install
  $(INSTALL_DIR) $(1)/$(FIRMWARE_PATH)
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fmacfw.bin $(1)/$(FIRMWARE_PATH)/fmacfw.
      bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fmacfw_rf.bin $(1)/$(FIRMWARE_PATH)/
      fmacfw_rf.bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fw_adid.bin $(1)/$(FIRMWARE_PATH)/fw_adid
      .bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fw_patch.bin $(1)/$(FIRMWARE_PATH)/
      fw_patch.bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fw_patch_bt_only.bin $(1)/$(
      FIRMWARE_PATH)/fw_patch_bt_only.bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fw_patch_combo.bin $(1)/$(FIRMWARE_PATH
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    )/fw_patch_combo.bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fw_patch_table.bin $(1)/$(FIRMWARE_PATH)/
    fw_patch_table.bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fw_patch_test.bin $(1)/$(FIRMWARE_PATH)/
    fw_patch_test.bin

endef
$(eval $(call BuildPackage,aic8800-firmware))
```

说明：可以看到整个 Makefile 文件就只是做了简单的拷贝动作。

说明

firmware 文件在编译后一般在系统 lib/firmware/下。需核对驱动中的定义是否一致，驱动版本不一致 PATH 的定义的位置和名称可能存在差异，按照驱动定义的实际 PATH 位置、名称修改路径。

```
aic880/aic8800_bsp/aic_bsp_driver.h
\#define AICBSP_FW_PATH          "/lib/firmware"

aic8800/aic8800_bsp/Makefile
CONFIG_AIC_FW_PATH = "/lib/firmware"

aic8800/aic8800_fdrv/Makefile
CONFIG_AIC_FW_PATH = "/lib/firmware"

aic8800/aic8800_fdrv/rwnx_platform.c
static const char *aic_fw_path = "/lib/firmware";
```

接着在 Tina4.0 根目录执行 m menuconfig 就可以看到新添加的 firmware 配置。

```
> Firmware
-*- aic8800-firmware.....................................AIC aic8800 firmware
```

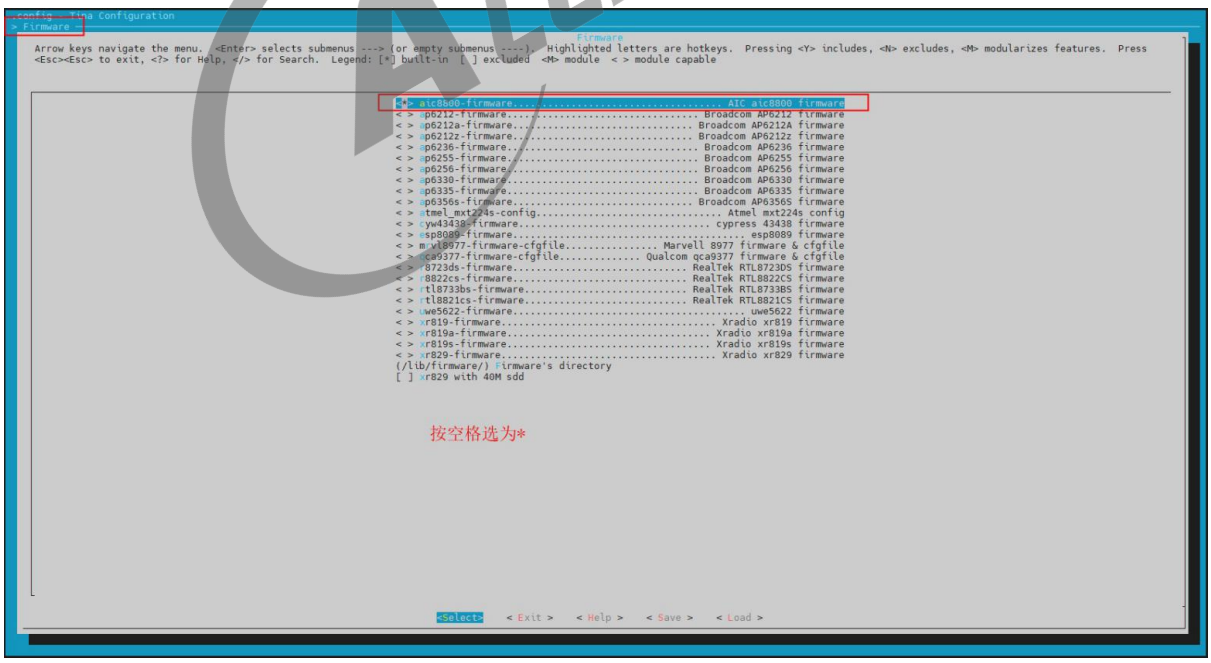


图 3-52: Tian-aic8800-firmware 配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3.4.1.5 应用工具适配

1. wifimanager 配置

```
make menuconfig
> Allwinner > wifimanager
```

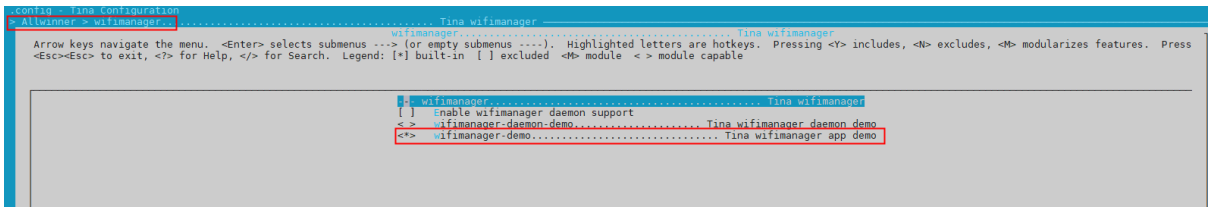


图 3-53: Tina-wifimanager 配置

2.smartlinkd 配网配置

```
make menuconfig
> Allwinner > smartlinkd
```

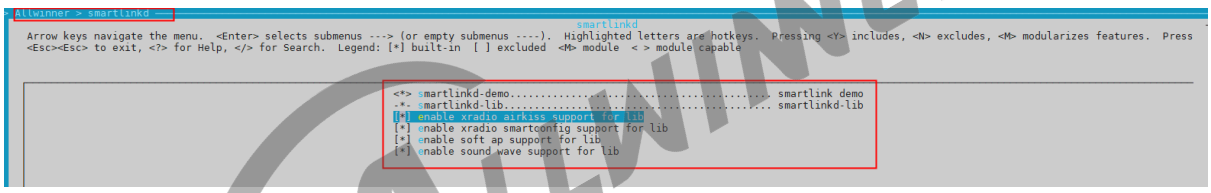


图 3-54: Tina-smartlinkd 配置

3. softap 配置

```
make menuconfig
> Allwinner > softap
```

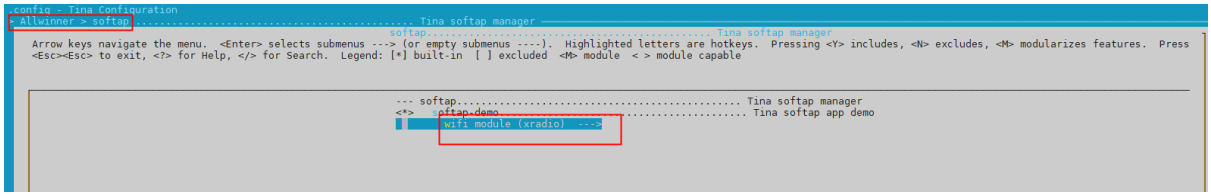


图 3-55: Tina-softap 配置

4. iperf 工具配置

```
make menuconfig
> Network
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

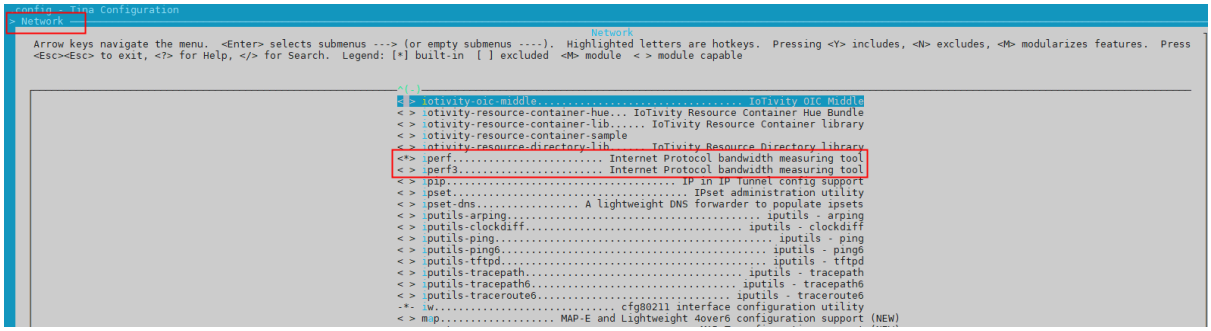


图 3-56: Tina-iperf 工具配置

5. WPA 配置

wpa 工具包括 wpa-supplicant 服务的和 wpa-cli 客户端。

```
make menuconfig
> Network
```

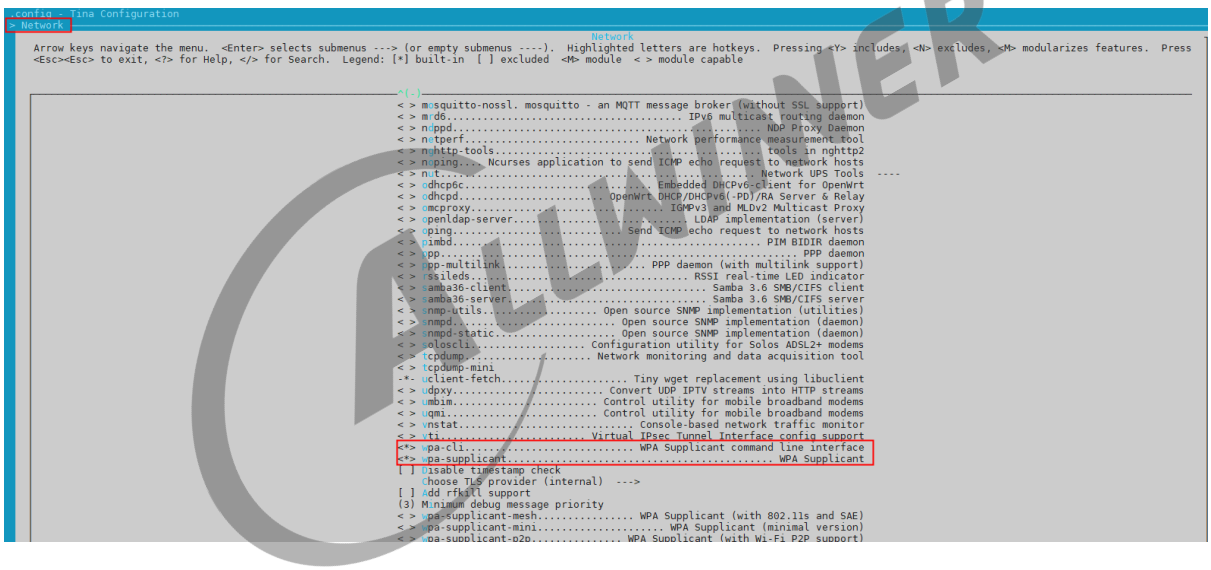


图 3-57: Tina-wpasupplicant 配置

3.4.2 AIC8800 USB 模组移植

主控：TV303

无线模组：AW869AU

内核版本：linux-5.4

方案：tv303c2_tuna_p3-userdebug

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3.4.2.1 内核驱动适配

内核驱动适配分为如下四个步骤：

- 获取源码。
- 添加 Kconfig 和 Makefile 配置。
- make kernel_menuconfig 配置。
- 编译。

1. 获取驱动源码，放到内核驱动路径下。客户可以从一号通账号获取。

```
longan/kernel/linux-5.4/drivers/net/wireless/aic8800
 ├──── aic8800_bsp
 ├──── aic8800_btlpm
 ├──── aic8800_btusb
 ├──── aic8800_fdrv
 ├──── Kconfig
 └──── Makefile
```

2. 内核添加 Kconfig 和 Makefile 的配置。

longan/kernel/linux-5.4/drivers/net/wireless/Kconfig 文件中引入 AW869AU 驱动的 Kconfig 配置。

```
source "drivers/net/wireless/aic8800/Kconfig"
```

longan/kernel/linux-5.4//drivers/net/wireless/Makefile 文件中引入 AW869B 驱动的 Makefile 配置。

```
obj-$(CONFIG_AIC_WLAN_SUPPORT)  += aic8800/
```

3.make kernel_menuconfig 配置

在 longan/kernel/linux-5.4/内核目录执行 m kernel_menuconfig。

a.AIC8800 驱动配置

```
> Device Drivers > Network device support > Wireless LAN
[*]  AIC wireless Support
     Enable Chip Interface (USB interface support)  --->
       ( ) SDIO interface support
       (X) USB interface support
[*]  Enable usb message endpoint (NEW)
<M>  AIC8800 wlan Support
< >  AIC8800 bluetooth Support (UART)
<M>  AIC8800 bluetooth Support (USB)
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

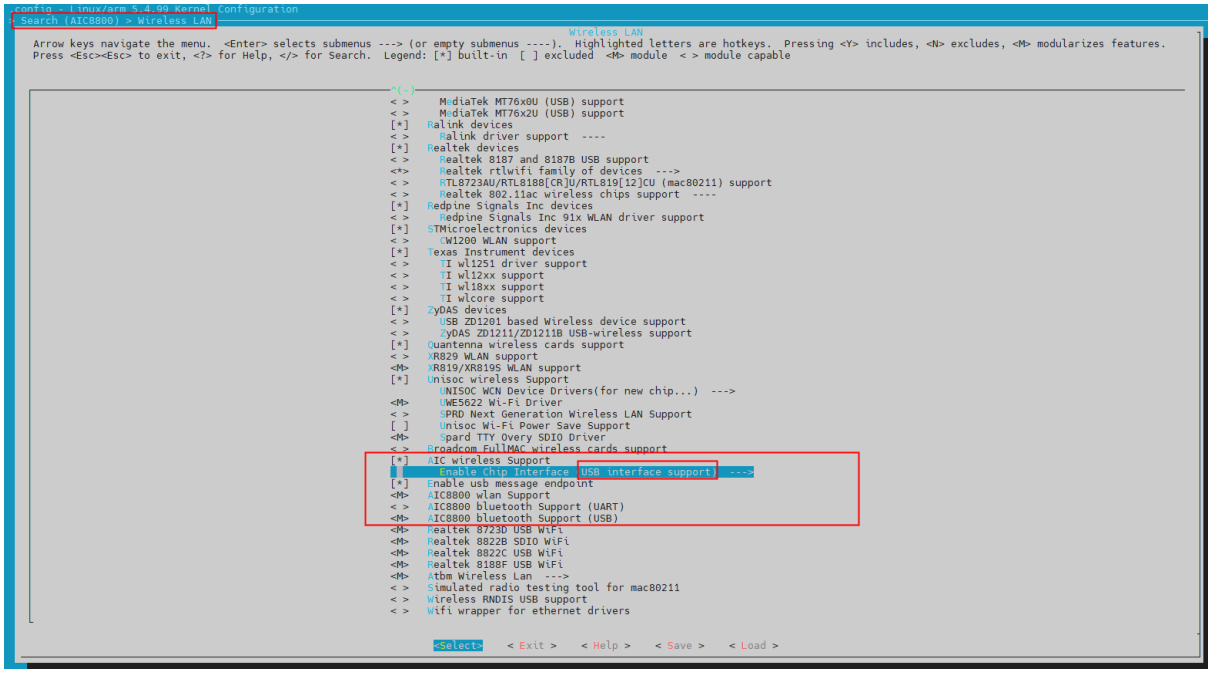


图 3-58: Tina-aic8800USB 内核配置

b.sunxi-rf 的配置

```
> Device Drivers > Misc devices
    <*> Allwinner rfkill driver
```

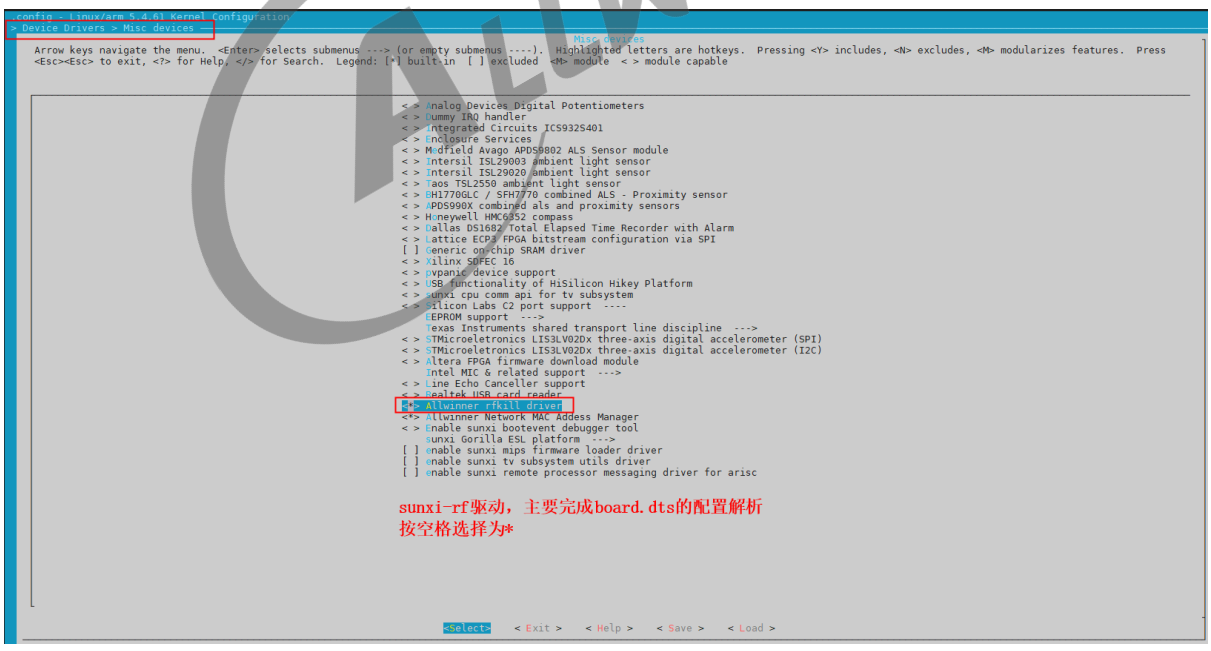


图 3-59: Tina-sunxi-rf 配置

c.USB 的配置

```
> Device Drivers > USB support
  -*-  Support for Host-side USB
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

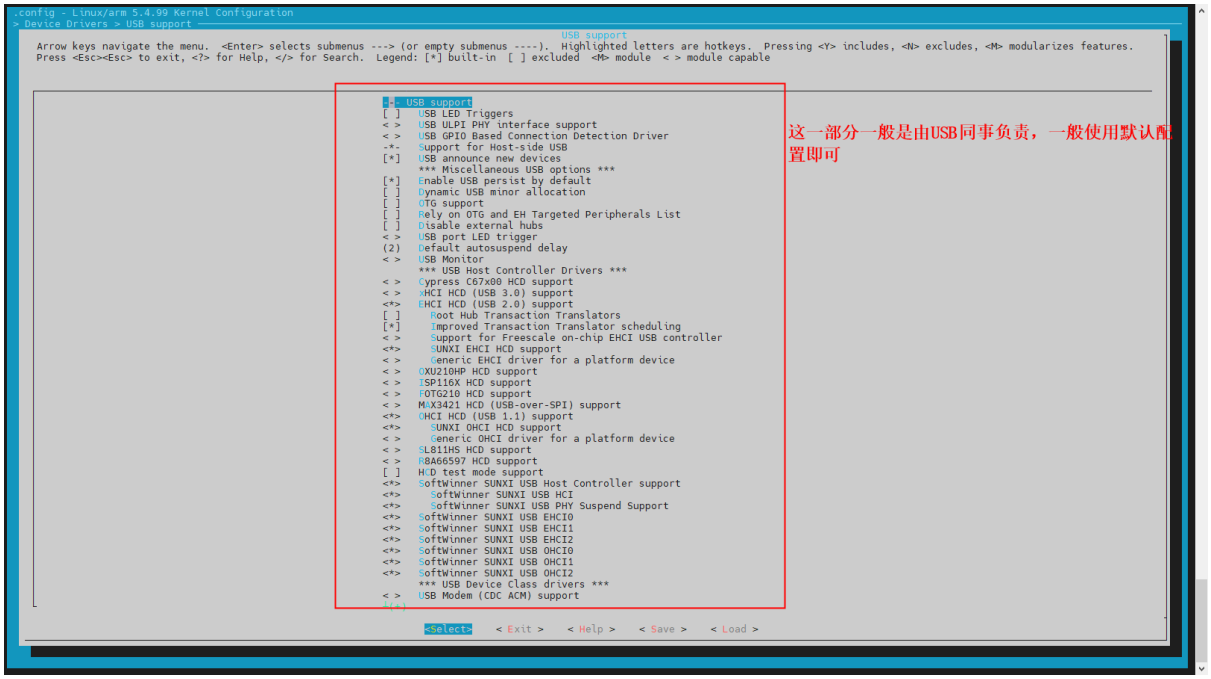


图 3-60: Tina-USB 配置

4. 编译

在 longan/kernel/linux-5.4/内核目录执行 mkernel 进行编译。

```
├──── aic8800_bsp
│    ├──── aic8800_bsp.ko
│    ├──── ...
├──── aic8800_btlpm
│    ├──── aic8800_btlpm.ko
│    ├──── ...
├──── aic8800_btusb
│    ├──── aic8800_btusb.ko
│    ├──── ...
├──── aic8800_fdrv
│    ├──── aic8800_fdrv.ko
│    ├──── ...
├──── built-in.a
├──── Kconfig
├──── Makefile
├──── modules.builtin
└──── modules.order
```

3.4.2.2 硬件资源适配

在 device/config/chips/tv303-c2/configs/tuna_p3/board.dts 中添加引脚配置。

```
&rfkill {
    compatible = "allwinner,sunxi-rfkill";
    chip_en;
    power_en;
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    pinctrl-0;
    pinctrl-names;
    status = "okay";

    /* wlan session */
    wlan {
        compatible   = "allwinner,sunxi-wlan";
        clocks;         /*时钟配置，USB模组一般无须配置，若有模组特殊要求，请参考SDIO模组配置*/
        clock-names;     /*时钟配置，USB模组一般无须配置，若有模组特殊要求，请参考SDIO模组配置*/
        wlan_power;      /*V303方案WiFi供电为常电无须软件配置开启*/
        wlan_power_vol;   /*V303方案WiFi供电为常电无须软件配置开启*/
        wlan_busnum;     /*USB模组无需配置*/
        wlan_regon   = <&r_pio PM 1 GPIO_ACTIVE_HIGH>;
        wlan_hostwake = <&r_pio PM 0 GPIO_ACTIVE_HIGH>; /*唤醒引脚配置，USB模组一般使用USB进行休眠唤醒，
      AIC8800 USB实际没有使用此引脚，可不用配置*/
        wakeup-source;
    };
    ...
};
```

属性	说明
clocks	用于配置使用主控提供的 32k 时钟
pinctrl-0	用于配置 pin 的复用功能
pinctrl-names	用于配置 pin state
wlan_busnum	表示 WiFi 所使用的 SDIO 控制器号
wlan_power	表示给 WiFi 模组供电的 regulator 名称
wlan_io_regulator	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_regon	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_hostwake	表示 WiFi 唤醒主控的 GPIO
chip_en	表示 WiFi 模组使能引脚，硬件未使用时不配置
power_en	表示模块外部的电源开关控制引脚

说明

以上所有项必须参看原理图进行配置，配置与原理图实际使用的资源保持一致；最好是和硬件同事一起确认，比如有些设计供电可能是没有 axp 的，硬件直接供电了，所以不需要配置。

3.4.2.3 方案 module 适配

在 target/allwinner/tv303-common/modules.mk 添加模块配置。

```
define KernelPackage/net-aic8800
 SUBMENU:=$(WIRELESS_MENU)
 TITLE:=aic8800 support (staging)
 DEPENDS:= +@IPV6
 KCONFIG:=\
  CONFIG_AIC8800_BTLPM_SUPPORT=m \
  CONFIG_AIC8800_WLAN_SUPPORT=m \
  CONFIG_AIC_WLAN_SUPPORT=m \
  CONFIG_PM=y\
  CONFIG_RFKILL=y \
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
  CONFIG_RFKILL_PM=y \
  CONFIG_RFKILL_GPIO=y

 FILES+=$(LINUX_DIR)/drivers/net/wireless/aic8800/aic8800_bsp/aic8800_bsp.ko
 FILES+=$(LINUX_DIR)/drivers/net/wireless/aic8800/aic8800_btlpm/aic8800_btlpm.ko
 FILES+=$(LINUX_DIR)/drivers/net/wireless/aic8800/aic8800_btusb/aic8800_btusb.ko
 FILES+=$(LINUX_DIR)/drivers/net/wireless/aic8800/aic8800_fdrv/aic8800_fdrv.ko

 AUTOLOAD:=$(call AutoProbe, aic8800_bsp aic8800_btlpm aic8800_btusb.ko aic8800_fdrv)
endef

define KernelPackage/net-aic8800/description
 Kernel modules for aic8800 support
endef

$(eval $(call KernelPackage,net-aic8800))
```

几点说明：

- DEPENDS：Tina 包依赖配置
- KCONFIG：内核依赖配置
- FILES：内核模块路径

一般不需要用户自己适配，可以自己从已经适配过的方案的 modules.mk 中拷贝过来就好，也可以直接拷贝这段代码。

接着在 Tina4.0 根目录执行 m menuconfig 就可以看到新添加的模组。

```
source build/envsetup.sh
lunch d1_nezha-tina  //根据实际方案选定

make menuconfig
> Kernel modules > Wireless Drivers
```

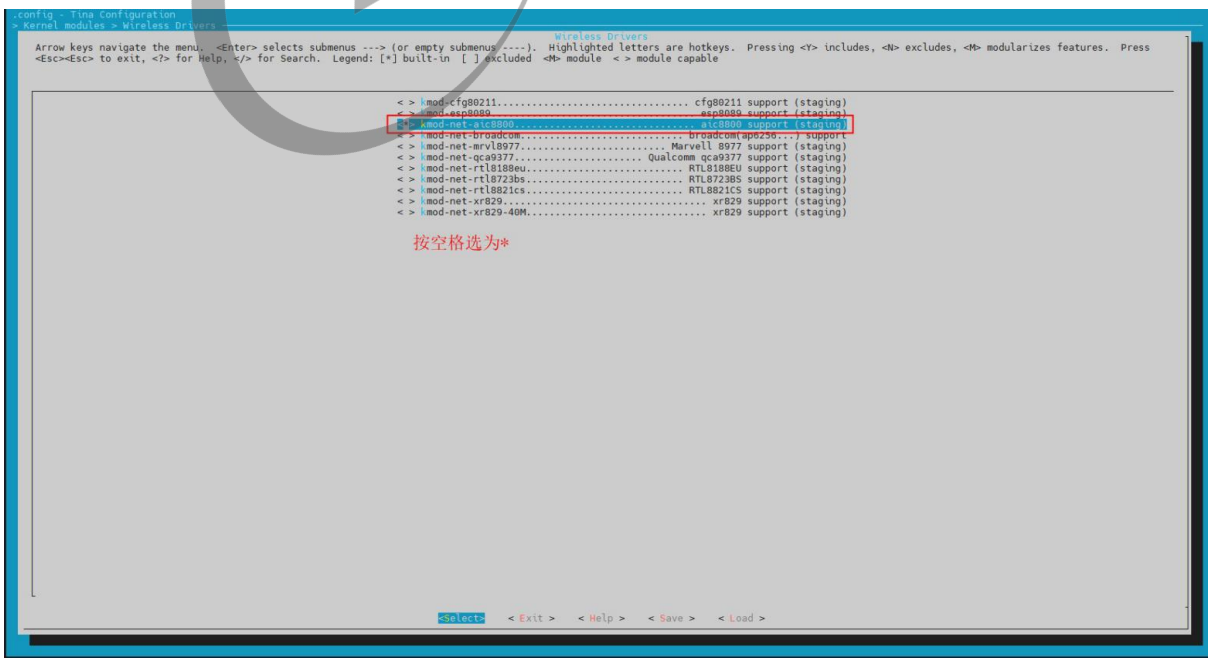


图 3-61: Tina-aic8800USB-module 配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

3.4.2.4 添加 Firmware

在 package/firmware/linux-firmware/添加 aic8800 需要的 firmware。

```
package/firmware/linux-firmware/aic8800
 ├──── aic8800.mk
 ├──── aic_userconfig.txt
 ├──── fmacfw.bin
 ├──── fmacfw_usb.bin
 ├──── fmacfw_rf.bin
 ├──── fmacfw_rf_usb.bin
 ├──── fw_adid.bin
 ├──── fw_patch.bin
 ├──── fw_patch_table.bin
 ├──── fw_adid_u03.bin
 ├──── fw_patch_u03.bin
 ├──── fw_patch_table_u03.bin
```

配置 aic8800.mk 文件

```
Package/aic8800-firmware = $(call Package/firmware-default,AIC aic8800 firmware)

define Package/aic8800-firmware/install
  $(INSTALL_DIR) $(1)/$(FIRMWARE_PATH)
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/aic_userconfig.txt $(1)/$(FIRMWARE_PATH)/
      aic_userconfig.txt
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fmacfw.bin $(1)/$(FIRMWARE_PATH)/fmacfw.
      bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fmacfw_usb.bin $(1)/$(FIRMWARE_PATH)/
      fmacfw_usb.bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fmacfw_rf.bin $(1)/$(FIRMWARE_PATH)/
      fmacfw_rf.bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fmacfw_rf_usb.bin $(1)/$(FIRMWARE_PATH)/
      fmacfw_rf_usb.bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fw_adid.bin $(1)/$(FIRMWARE_PATH)/fw_adid
      .bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fw_patch.bin $(1)/$(FIRMWARE_PATH)/
      fw_patch.bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fw_patch_table.bin $(1)/$(FIRMWARE_PATH)/
      fw_patch_table.bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fw_adid_u03.bin $(1)/$(FIRMWARE_PATH)/
      fw_adid_u03.bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fw_patch_u03.bin $(1)/$(FIRMWARE_PATH)/
      fw_patch_u03.bin
  $(INSTALL_DATA) $(TOPDIR)/package/firmware/linux-firmware/aic8800/fw_patch_table_u03.bin $(1)/$(
      FIRMWARE_PATH)/fw_patch_table_u03.bin

endef
$(eval $(call BuildPackage,aic8800-firmware))
```

说明：可以看到整个 Makefile 文件就只是做了简单的拷贝动作。

说明

firmware 文件在编译后一般在系统 lib/firmware/。AIC 驱动使用 request_firmware 函数在内核定义的 firmware 路径中查找对应名字，一般是在 lib/firmware/下查找，驱动版本不一致 PATH 的定义的位置和名称可能存在差异，按照驱动定义的实际 PATH 位置、名称修改路径。

接着在 Tina4.0 根目录执行 m menuconfig 就可以看到新添加的 firmware 配置。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

> Firmware
-*- aic8800-firmware.....AIC aic8800 firmware

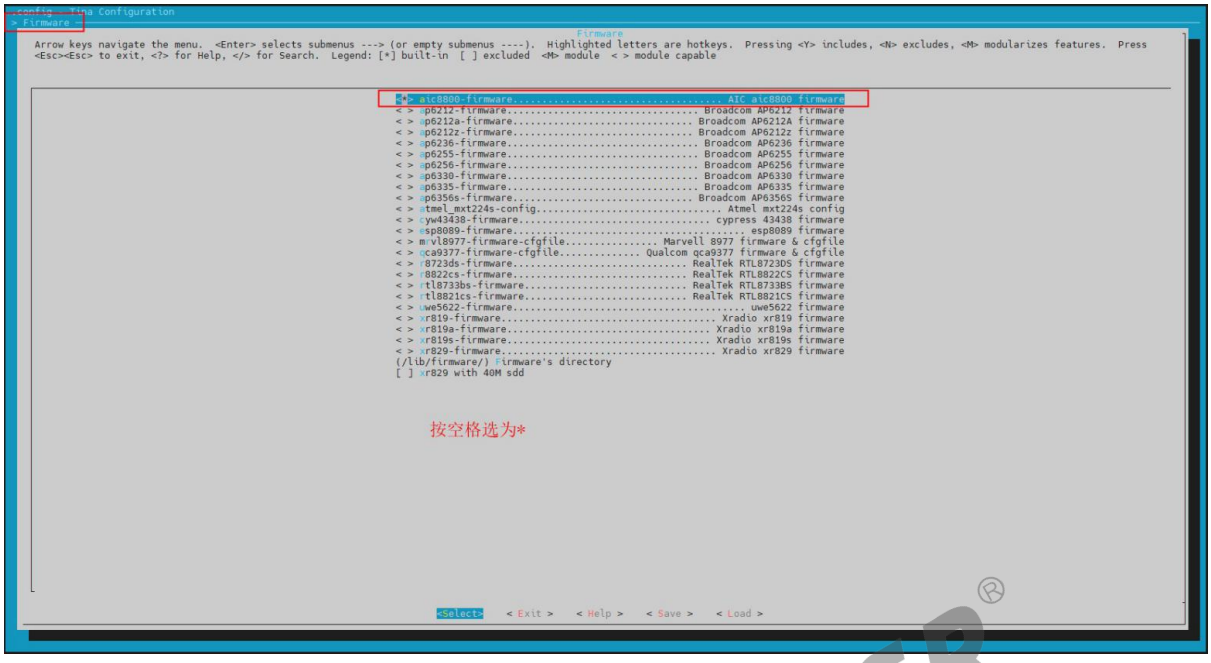


图 3-62: Tian-aic8800USB-firmware 配置

3.4.2.5 应用工具的适配

1.wifimanager 配置

make menuconfig
> Allwinner > wifimanager

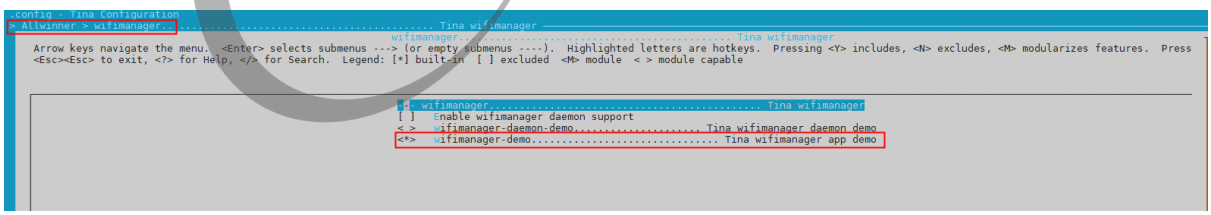


图 3-63: Tina-wifimanager 配置

2.smartlinkd 配网配置

make menuconfig
> Allwinner > smartlinkd

版权所有 © 珠海全志科技股份有限公司。保留一切权利

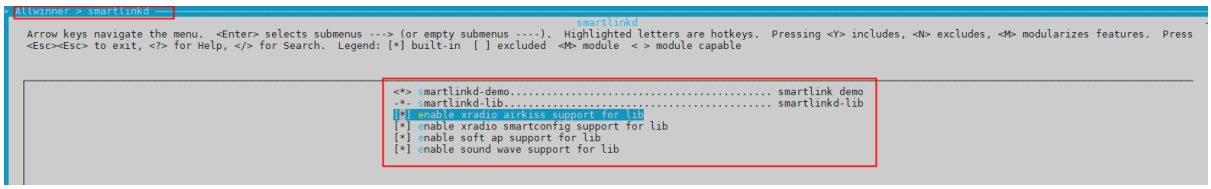


图 3-64: Tina-smartlinkd 配置

3.softap 配置

make menuconfig
> Allwinner > softap

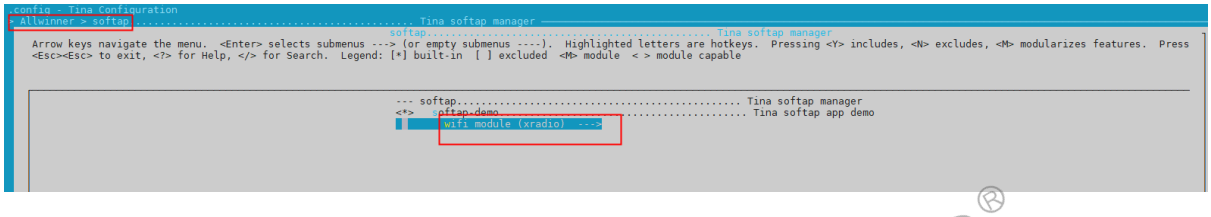


图 3-65: Tina-softap 配置

4.iperf 工具配置

make menuconfig
> Network

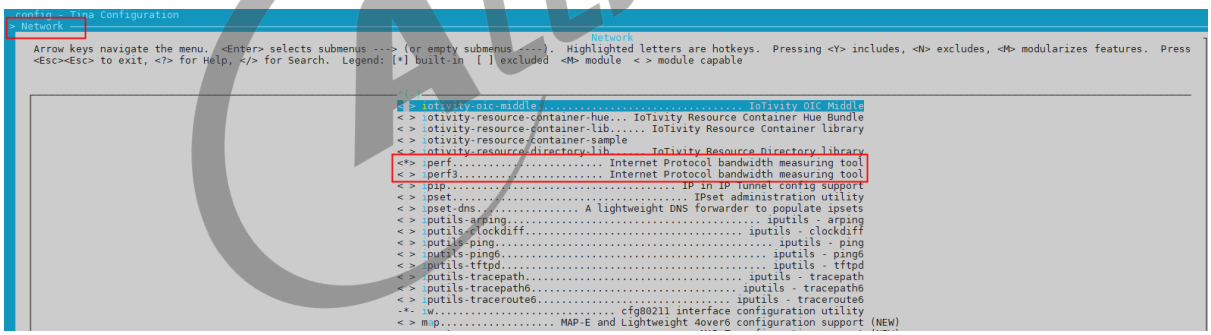


图 3-66: Tina-iperf 工具配置

5.WPA 配置

wpa 工具包括 wpa-supplicant 服务的和 wpa-cli 客户端。

make menuconfig
> Network

版权所有 © 珠海全志科技股份有限公司。保留一切权利

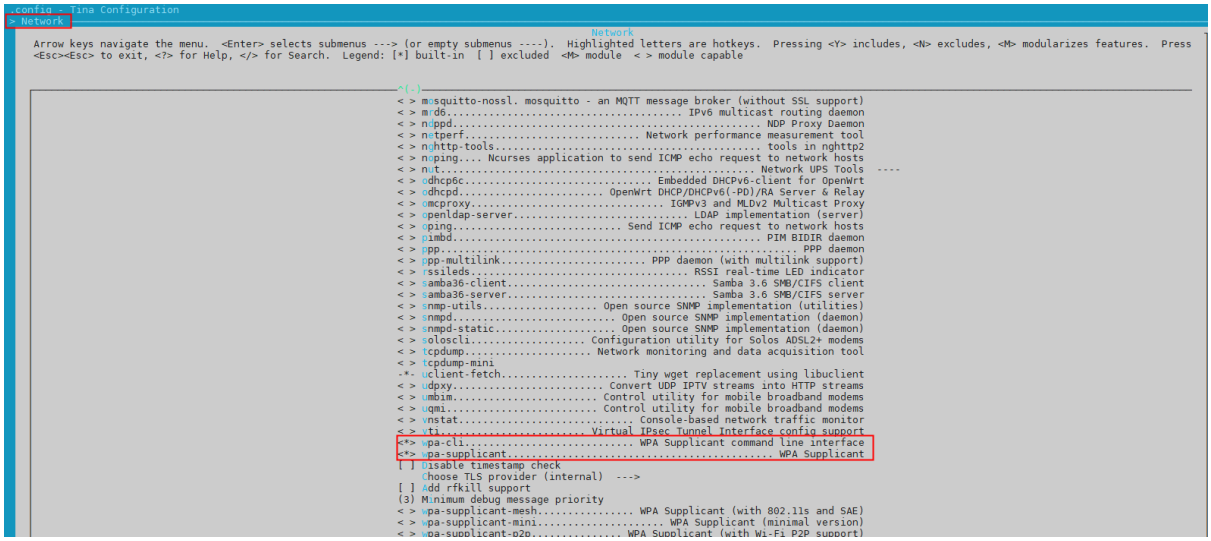


图 3-67: Tina-wpasupplicant 配置

3.4.3 AIC8800D80/D40/D40L 模组移植

主控：AI985

无线模组：AIC8800D80/D40/D40L

内核版本：linux-5.15

方案：ai985-scanp_fastboot-Tina

Tina 版本：Tina-5.0

说明

该款 Wifi 模组 IO 电压要求 1.8V

3.4.3.1 内核驱动适配

内核驱动适配分为如下四个步骤：

- 获取源码。
- 添加 Kconfig 和 Makefile 配置。
- make kernel_menuconfig 配置。
- 编译。

1）获取驱动源码，放到内核驱动路径下。客户可以从一号通账号获取。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
tina-5.0/bsp/drivers/net/wireless/aic8800
├──── aic8800_bsp
├──── aic8800_btlpm
├──── aic8800_fdrv
├──── Kconfig
└──── Makefile
```

2）内核添加 Kconfig 和 Makefile 的配置。

tina-5.0/bsp/drivers/net/wireless/Kconfig 文件中引入 AIC8800D80/D40/D40L 驱动的 Kconfig 配置。

```
source "$(BSP_TOP)drivers/net/wireless/aic8800/Kconfig"
```

tina-5.0/bsp/drivers/net/wireless/Makefile 文件中引入 AIC8800D80/D40/D40L 驱动的 Makefile 配置。

```
obj-$(CONFIG_AIC_WLAN_SUPPORT) += aic8800/
```

3）make kernel_menuconfig 配置

在 Tina-5.0 根目录执行 m kernel_menuconfig。

a. aic8800 驱动的配置

```
> Allwinner BSP > Device Drivers > Network Device Drivers > Wireless LAN
   <*> AIC wireless Support
   <M>  AIC8800 wlan Support
   <M>  AIC8800 bluetooth Support
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

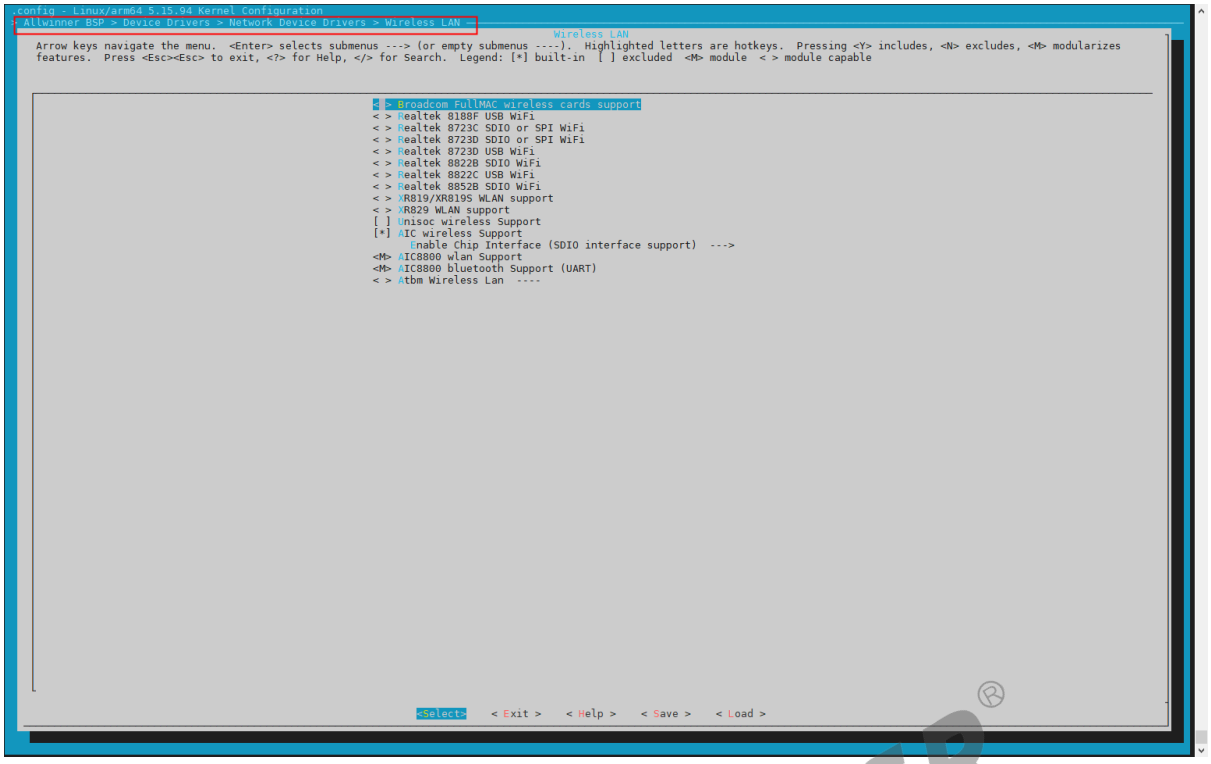


图 3-68: Tina-5.0-aic8800 内核配置

b. sunxi-rf 的配置

Allwinner BSP > Device Drivers > Misc Devices Drivers
<M> Allwinner rfkill driver

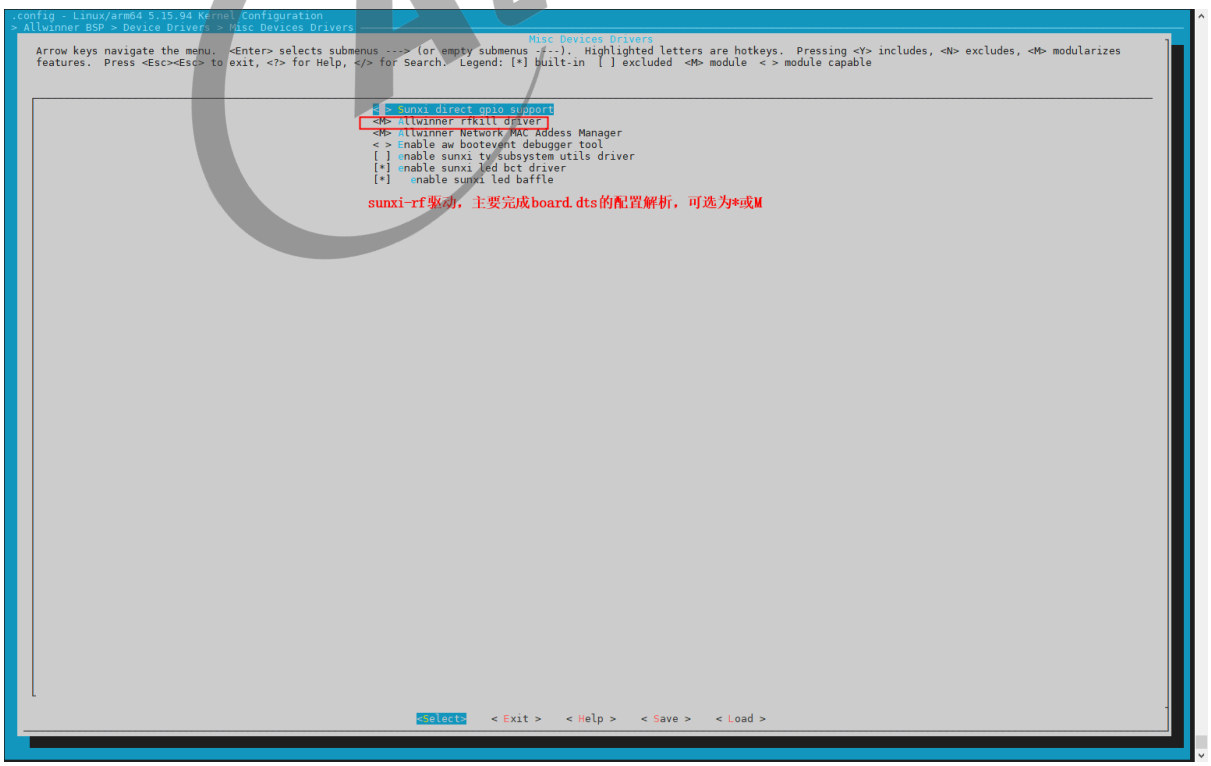


图 3-69: Tina-5.0-sunxi-rf 配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

c. SDIO 的配置

```
> Allwinner BSP > Device Drivers > SD/MMC Drivers
    <*> Allwinner SD/MMC Host Controller Support
```

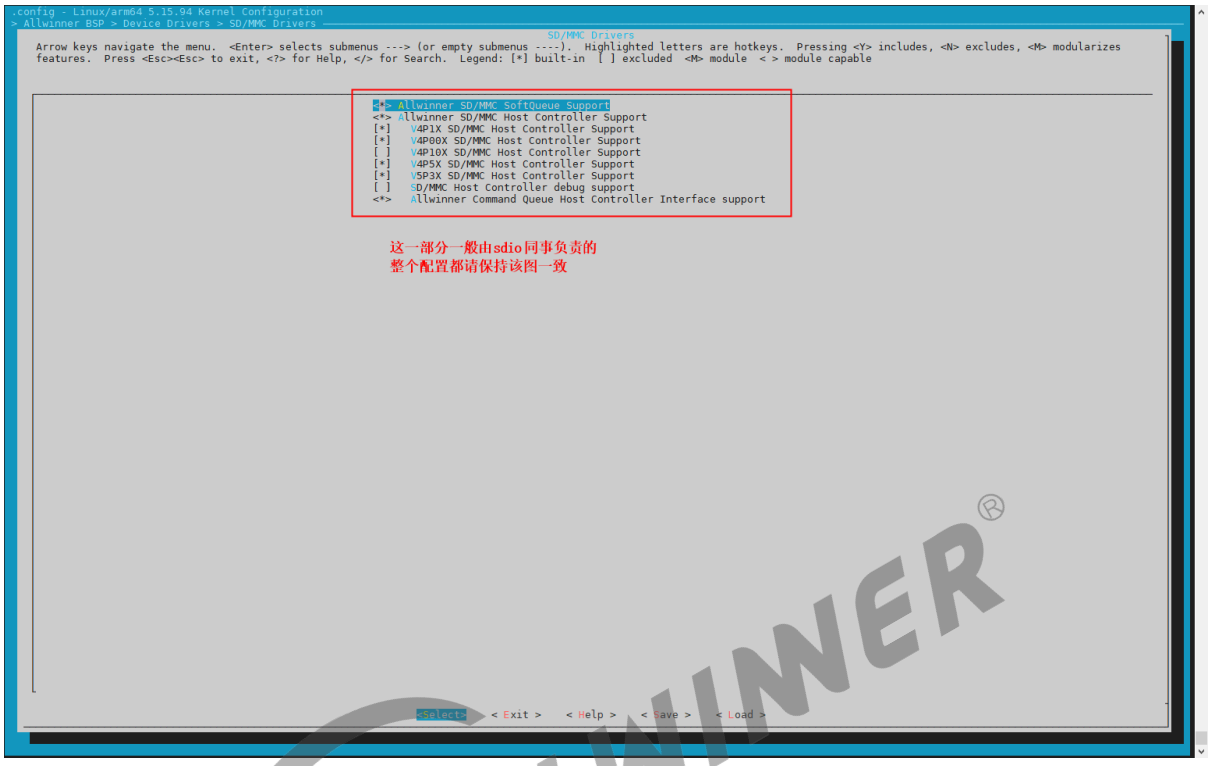


图 3-70: Tina-5.0-sdc 配置

3.4.3.2 硬件资源适配

在 tina-5.0/device/config/chips/ai985/configs/scanp_fastboot/linux-5.15/board.dts 中添加引脚配置。

```
&pio {
    vcc-pg-supply = <&reg_pio1_8>; /*AIC8800的IO电压为1.8V，这里需要配置PG口1.8V*/
    vcc-pc-supply = <&reg_pio1_8>;
    vcc-pf-supply = <&reg_pio1_8>;
    vcc-pe-supply = <&reg_pio1_8>;
    vcc-pfo-supply = <&reg_pio3_3>;
    sdc0_pins_a: sdc0@0 {
        ...
    };
  ...
    rfkill {
      compatible = "allwinner,sunxi-rfkill";
      chip_en;
      power_en = <&r_pio PL 7 GPIO_ACTIVE_HIGH>;
      pinctrl-0;
      pinctrl-names;
      status = "okay";
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
        /* wlan session */
        wlan {
          compatible   = "allwinner,sunxi-wlan";
          clocks       = <&rtc_ccu CLK_DCXO24M_OUT>, <&rtc_ccu CLK_OSC32K_OUT>;
          clock-names  = "dcxo24M-out", "osc32k-out";
          wlan_power   = "axp2202-aldo3", "axp2202-bldo1", "axp2202-bldo2";
          wlan_power_vol= <3300000>, <1800000>, <1800000>;
          wlan_busnum  = <0x1>;
          wlan_regon   = <&r_pio PM 1 GPIO_ACTIVE_HIGH>;
          wlan_hostwake = <&r_pio PM 0 GPIO_ACTIVE_HIGH>;
          wakeup-source;
          regulator-boot-on;
        };
        ...
    }

...
```

属性	说明
wlan_busnum	表示 WiFi 所使用的 SDIO 控制器号
wlan_power	表示给 WiFi 模组供电的 regulator 名称
wlan_power_vol	表示 WiFi 模组需要使用到的电的电压
wlan_regon	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_hostwake	表示 WiFi 唤醒主控的 GPIO
chip_en	表示 WiFi 模组使能引脚，硬件未使用时不配置
power_en	表示模块外部的电源开关控制引脚

说明

以上所有项必须参看原理图进行配置，配置与原理图实际使用的资源保持一致；最好是和硬件同事一起确认，比如有些设计供电可能是没有 axp 的，硬件直接供电了，所以不需要配置，特别注意的就是 sdio 的配置。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

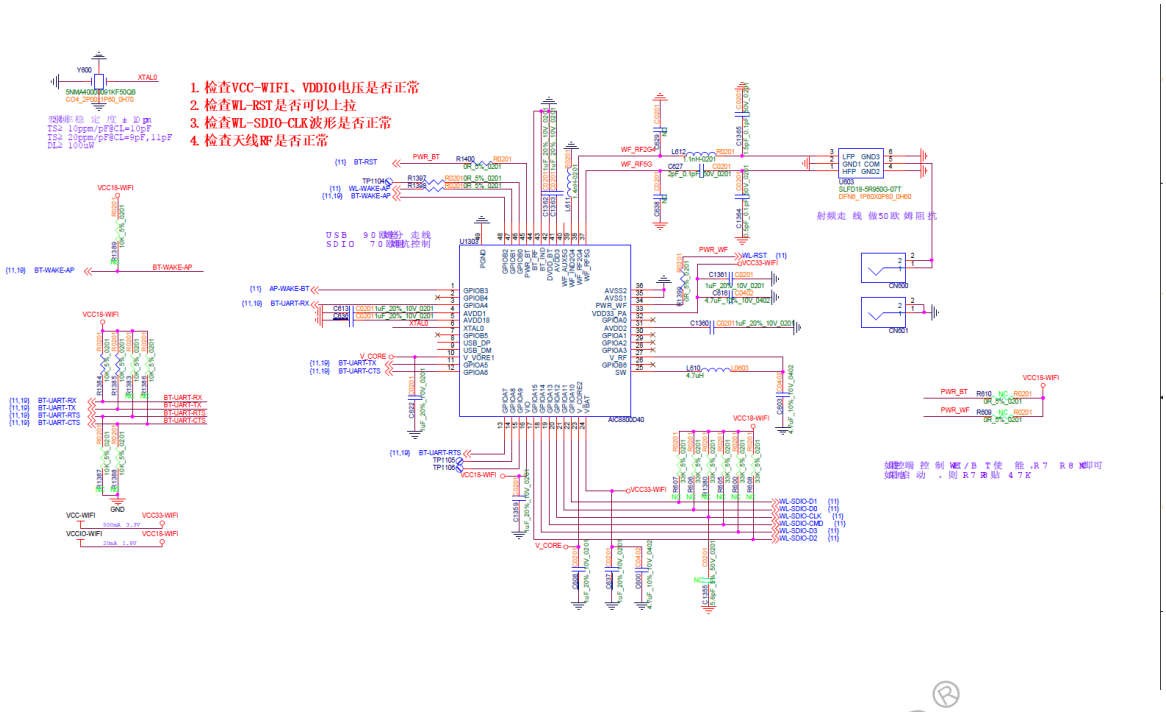


图 3-71: AIC8800D40 与主控连接原理图

3.4.3.3 方案 module 适配

在 tina-5.0/openwrt/target/ai985/ai985-common 添加模块配置。

```
define KernelPackage/net-aic8800
  SUBMENU:=$(WIRELESS_MENU)
  TITLE:=aic8800 support (staging)
  DEPENDS:= +aic8800-firmware +@IPV6 +@PACKAGE_aic8800-rftest
  KCONFIG:=\
  CONFIG_AIC_WLAN_SUPPORT=y\
  CONFIG_AIC8800_WLAN_SUPPORT=m\
  CONFIG_AIC8800_BTLPM_SUPPORT=m\
  CONFIG_AW_RFKILL=y\
  CONFIG_AW_MACADDR_MGT=y\
  CONFIG_BT=y\
  CONFIG_BT_RFCOMM=y\
  CONFIG_BT_RFCOMM_TTY=y\
  CONFIG_BT_HCIUART=y\
  CONFIG_BT_HCIUART_H4=y\
  CONFIG_RFKILL=y\
  CONFIG_RFKILL_GPIO=y
   FILES:=$(LICHEE_OUT_DIR)/$(LICHEE_IC)/kernel/build/bsp/drivers/net/wireless/aic8800/aic8800_bsp/aic8800_bsp.
       ko
  FILES+=$(LICHEE_OUT_DIR)/$(LICHEE_IC)/kernel/build/bsp/drivers/net/wireless/aic8800/aic8800_btlpm/
       aic8800_btlpm.ko
  FILES+=$(LICHEE_OUT_DIR)/$(LICHEE_IC)/kernel/build/bsp/drivers/net/wireless/aic8800/aic8800_fdrv/aic8800_fdrv.
       ko
  AUTOLOAD:=$(call AutoProbe, aic8800_bsp aic8800_btlpm aic8800_fdrv)
 endef

 define KernelPackage/net-aic8800/description
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
 Kernel modules for aic8800 support
endef

$(eval $(call KernelPackage,net-aic8800))
```

几点说明：

- DEPENDS：Tina 包依赖配置
- KCONFIG：内核依赖配置
- FILES：内核模块路径

一般不需要用户自己适配，可以自己从已经适配过的方案的 modules.mk 中拷贝过来就好，也可以直接拷贝这段代码。

接着在 Tina5.0 根目录执行 m menuconfig 就可以看到新添加的模组。

```
source build/envsetup.sh
lunch ai985-scanp_fastboot  //根据实际方案选定

make menuconfig
> Kernel modules > Wireless Drivers
```

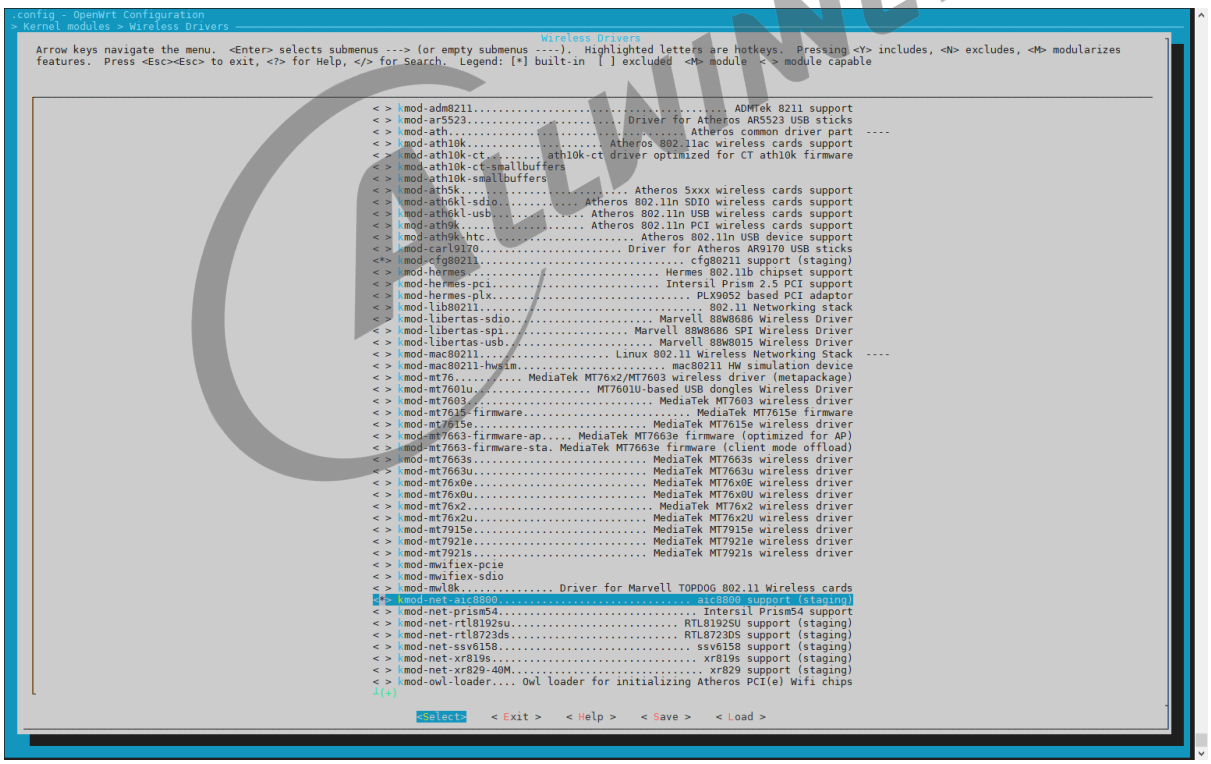


图 3-72: Tina-5.0-aic8800-module 配置

3.4.3.4 添加 Firmware

在 tina-5.0/platform/allwinner/wireless/firmware/aic8800 添加 aic8800 需要的 firmware。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

说明

aic8800/aic8800d80/aic8800d40/aic8800d40L 使用相同的驱动，但固件不一样
aic8800 使用 tina-5.0/platform/allwinner/wireless/firmware/aic8800 目录下的固件
aic8800d80/aic8800d40/aic8800d40L 使用 tina-5.0/platform/allwinner/wireless/firmware/aic8800/aic8800d80 目录下的固件

```
tina-5.0/platform/allwinner/wireless/firmware/aic8800
├── aic8800d80
│   ├── aic_userconfig_8800d80-Typ.txt
│   ├── fmacfw_8800d80.bin
│   ├── fmacfw_8800d80_u02.bin
│   ├── fmacfw_rf_8800d80.bin
│   ├── fmacfw_rf_8800d80_u02.bin
│   ├── fw_adid_8800d80.bin
│   ├── fw_adid_8800d80_u02.bin
│   ├── fw_patch_8800d80.bin
│   ├── fw_patch_8800d80_u02.bin
│   ├── fw_patch_table_8800d80.bin
│   ├── fw_patch_table_8800d80_u02.bin
│   ├── lmacfw_rf_8800d80.bin
│   └── lmacfw_rf_8800d80_u02.bin
├── aic_userconfig.txt
├── fmacfw.bin
├── fmacfw_rf.bin
├── fmacfw_rf_usb.bin
├── fmacfw_usb.bin
├── fw_adid.bin
├── fw_adid_u03.bin
├── fw_patch.bin
├── fw_patch_table.bin
├── fw_patch_table_u03.bin
└── fw_patch_u03.bin
```

修改 tina-5.0/openwrt/openwrt/package/firmware/linux-firmware 配置 aic8800.mk 文件

```
Package/aic8800-firmware = $(call Package/firmware-default,AIC aic8800 firmware)

SRC_CODE_DIR:=$(AIC8800_FW)

define Package/aic8800-firmware/install
  $(INSTALL_DIR) $(1)/$(FIRMWARE_PATH)
  $(INSTALL_DIR) $(1)/$(FIRMWARE_PATH)aic8800d80/

  $(INSTALL_DATA) $(SRC_CODE_DIR)/*.* $(1)/$(FIRMWARE_PATH)
  $(INSTALL_DATA) $(SRC_CODE_DIR)/aic8800d80/*.* $(1)/$(FIRMWARE_PATH)aic8800d80
endef
$(eval $(call BuildPackage,aic8800-firmware))
```

说明：可以看到整个 Makefile 文件就只是做了简单的拷贝动作。

说明

firmware 文件在编译后一般在系统 lib/firmware/。需核对驱动中的定义是否一致，驱动版本不一致 PATH 的定义的位置和名称可能存在差异，按照驱动定义的实际 PATH 位置、名称修改路径。

```
aic880/aic8800_bsp/aic_bsp_driver.h
\#define AICBSP_FW_PATH          "/lib/firmware"

aic8800/aic8800_bsp/Makefile
CONFIG_AIC_FW_PATH = "/lib/firmware"
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
aic8800/aic8800_fdrv/Makefile
CONFIG_AIC_FW_PATH = "/lib/firmware"

aic8800/aic8800_fdrv/rwnx_platform.c
static const char *aic_fw_path = "/lib/firmware";
```

接着在 Tina5.0 根目录执行 m menuconfig 就可以看到新添加的 firmware 配置。

```
> Firmware
-*- aic8800-firmware.....AIC aic8800 firmware
```

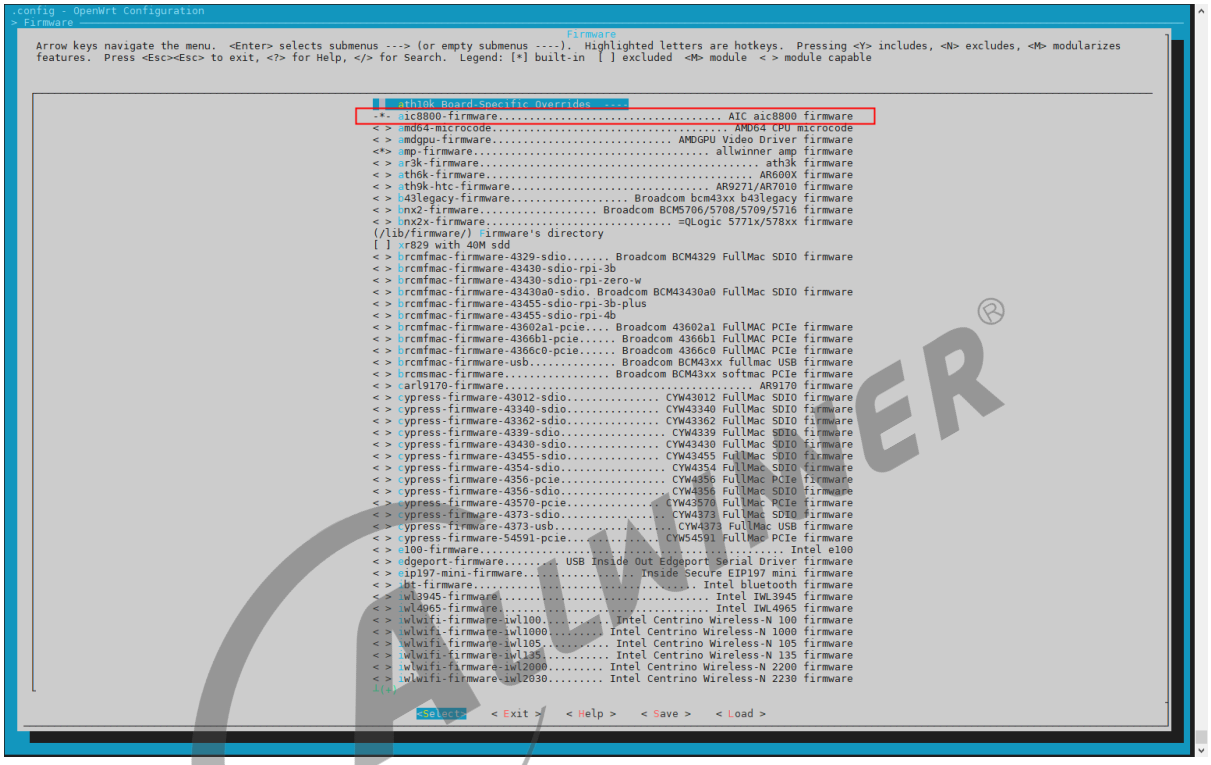


图 3-73: Tian-5.0-aic8800-firmware 配置

3.4.3.5 应用工具适配

1. wifimanager 配置

```
make menuconfig
> Allwinner > Wireless > wifimanager
```

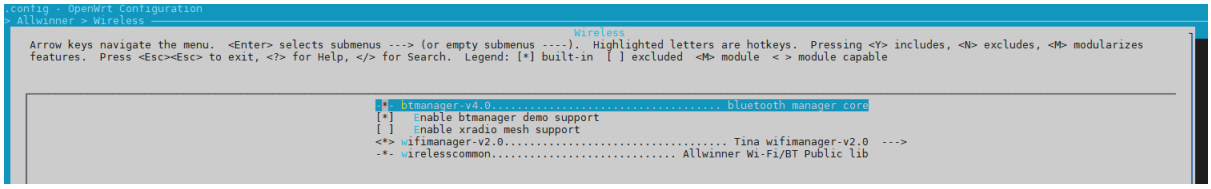


图 3-74: Tina-5.0-wifimanager 配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

2.iperf 工具配置

```
make menuconfig
> Network
```

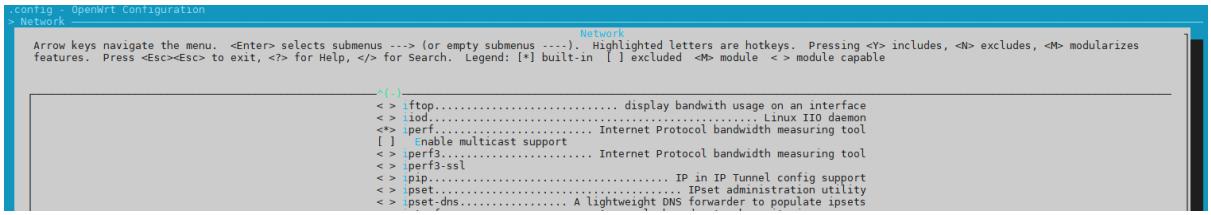


图 3-75: Tina-5.0-iperf 工具配置

3.WPA 配置

wpa 工具包括 wpa-supplicant 服务的和 wpa-cli 客户端。

```
make menuconfig
> Network > WirelessAPD
```

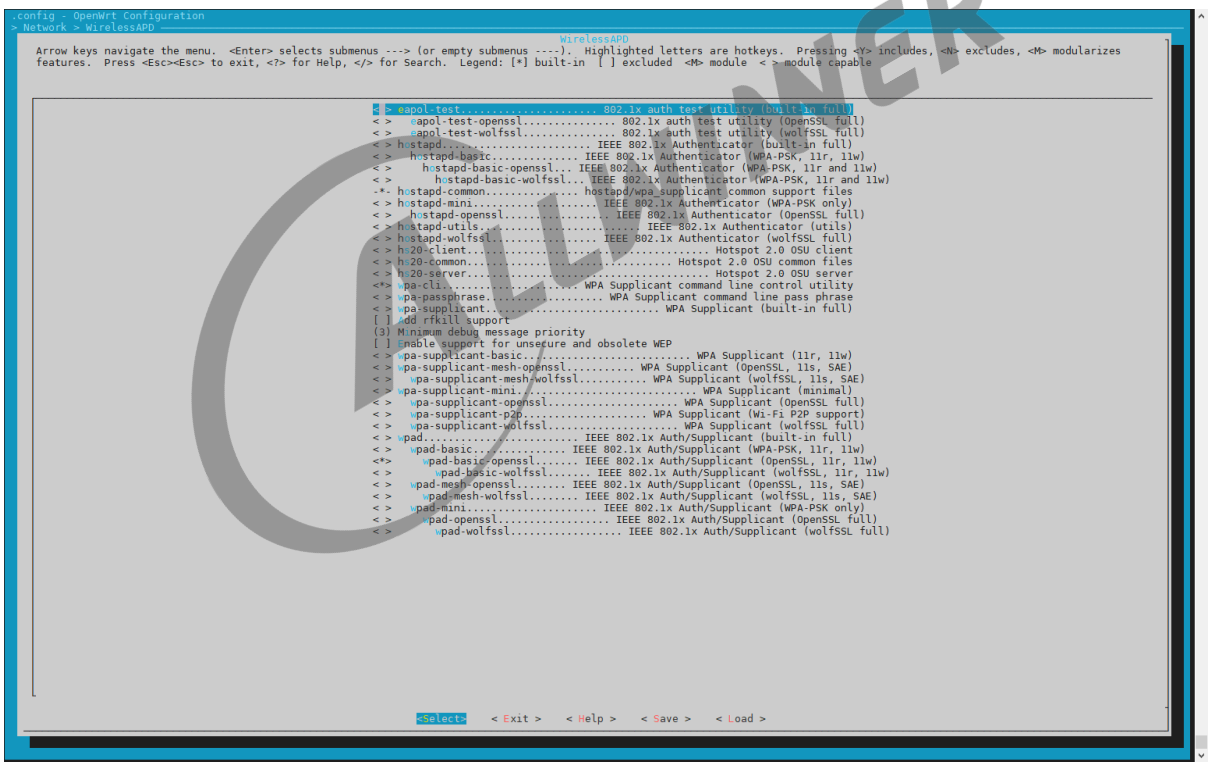


图 3-76: Tina-5.0-wpasupplicant 配置

3.4.4 AIC8800D40/D80 USB 模组移植

主控：H135

无线模组：AIC8800D40 USB Version

版权所有 © 珠海全志科技股份有限公司。保留一切权利

内核版本：linux-6.6

方案：h135-p1

Tina 版本：Tina5.0

3.4.4.1 内核驱动适配

全志后续需要将 AIC 的 SDIO 驱动和 USB 驱动合并为一个驱动。目前正在过渡阶段，将过渡阶段驱动暂时作为一个新驱动加入到 bsp 仓库中。过渡阶段将二合一驱动命名为 “aic_usb_sdio”。

内核驱动适配分为如下四个步骤：

- 获取源码。
- 添加 Kconfig 和 Makefile 配置。
- make kernel_menuconfig 配置。
- 编译。

1. 获取驱动源码，放到内核驱动路径下。客户可以从一号通账号获取。

```
tina5.0/bsp/drivers/net/wireless/aic_usb_sdio
├──── aic8800_bsp
├──── aic8800_btlpm
├──── aic8800_btusb
├──── aic8800_fdrv
├──── Kconfig
└──── Makefile
```

如果是在 tina4.0 上使用对应的路径是：

```
Tina4.0/lichee/linux-5.4/drivers/net/wireless/aic_usb_sdio
```

2. 内核添加 Kconfig 和 Makefile 的配置。

tina5.0/bsp/drivers/net/wireless/Kconfig 文件中引入 aic_usb_sdio 驱动的 Kconfig 配置。

```
source "$(BSP_TOP)drivers/net/wireless/aic_usb_sdio/Kconfig"
```

tina5.0/bsp/drivers/net/wireless/Makefile 文件中引入 aic_usb_sdio 驱动的 Makefile 配置。

```
obj-$(CONFIG_AIC_WLAN_SUPPORT) += aic_usb_sdio/
```

在 tina4.0 中的路径可参考 AIC8800 模组移植目录。

3.make kernel_menuconfig 配置

```
[*] AIC wireless Support
      Enable Chip Interface (USB interface support)  --->
    Choice host wake IRQ type (UNSET IRQ type use default config)  --->
[*] Enable usb message endpoint
<M> AIC8800 wlan Support
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
< > AIC8800 bluetooth Support (UART)
< > AIC8800 bluetooth Support (USB)
```

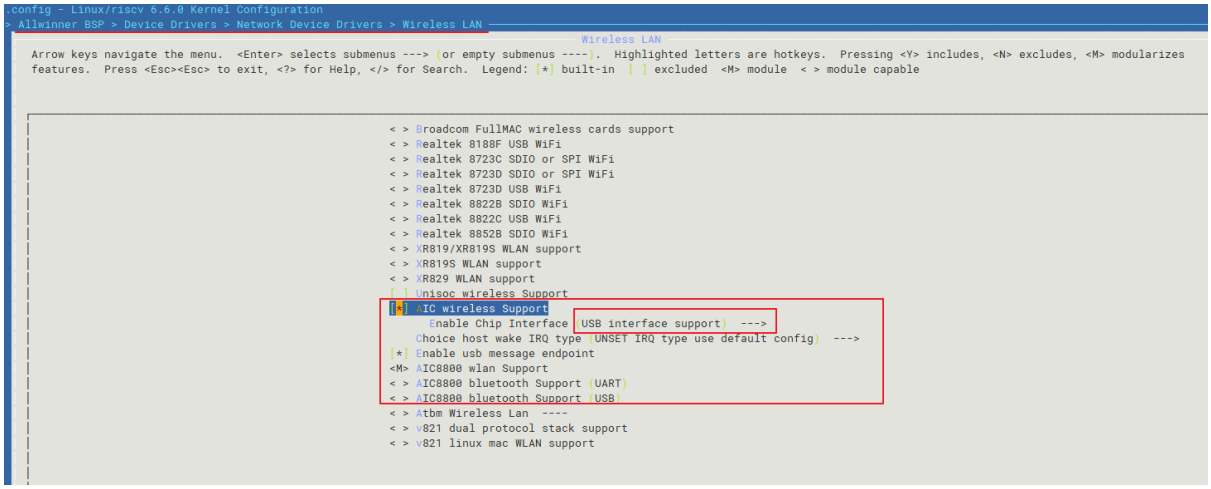


图 3-77: Tina5.0-aic_usb_sdio 内核配置

b.sunxi-rf 的配置

```
> Device Drivers > Misc devices
   <*> Allwinner rfkill driver
```

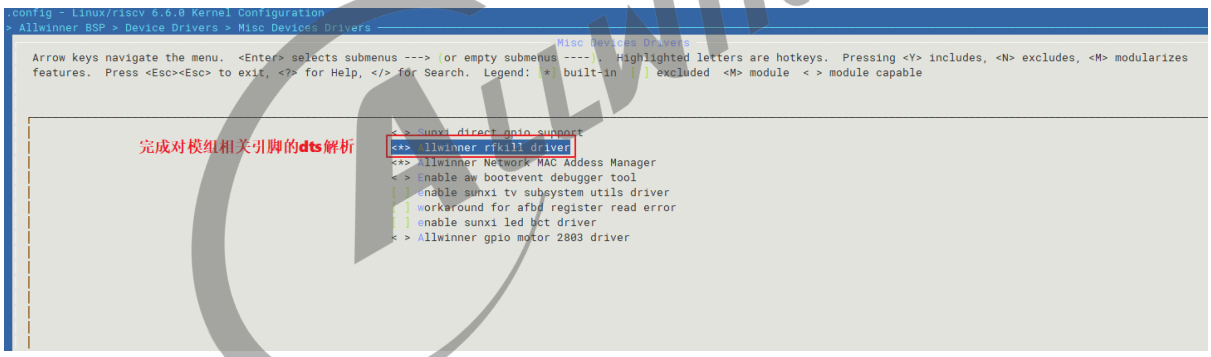


图 3-78: Tina5.0-sunxi-rfkill 内核配置

c.USB 的配置

```
> Device Drivers > USB support
  -*-  Support for Host-side USB
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

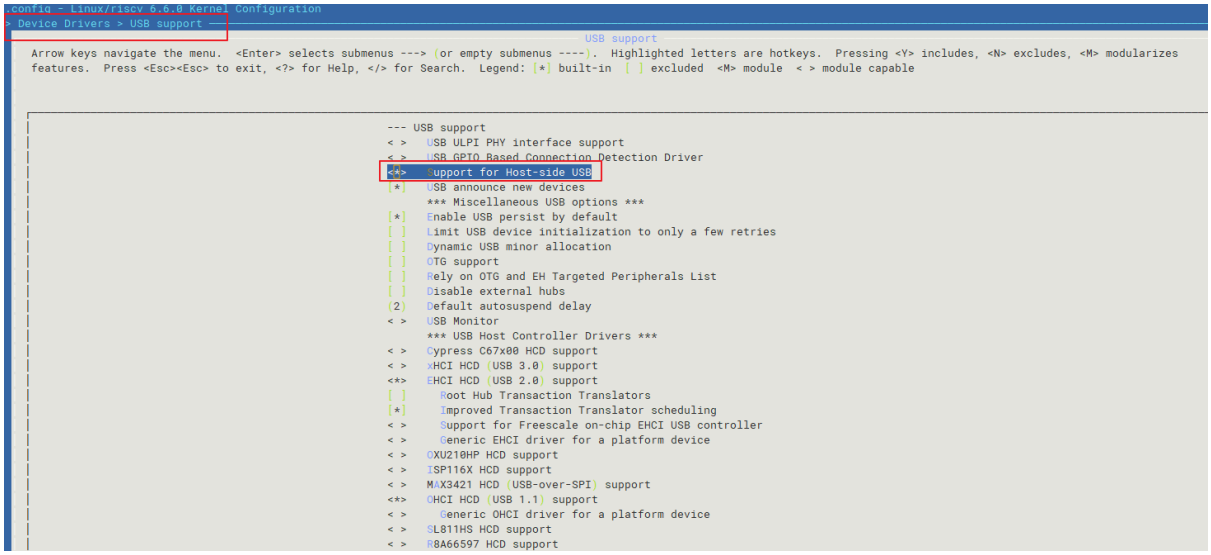


图 3-79: Tina5.0-USB 内核配置

4. 编译

在任意目录执行 mkernel 进行编译。

```
mkernel

tina5.0/out/h135/kernel/build/bsp/drivers/net/wireless/aic_usb_sdio
 ├── aic8800_bsp
 │   ├── aic8800_bsp.ko
 │   └── ...
 ├── aic8800_fdrv
 │   ├── aic8800_fdrv.ko
 │   └── ...
 ├── built-in.a
 └── modules.order
```

3.4.4.2 硬件资源适配

在 tina5.0/device/config/chips/h135/configs/p1/board.dts 中添加引脚配置。

```
...
&soc {
    ...
    rfkill: rfkill@0 {
        compatible = "allwinner,sunxi-rfkill";
        chip_en;
        power_en;
        pinctrl-0;
        pinctrl-names;
        status= "okay";
        wlan: wlan@0 {
            compatible   = "allwinner,sunxi-wlan";
            wlan_busnum   = <0x1>;
            wlan_regon   = <&pio PG 11 GPIO_ACTIVE_HIGH>;
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
                /*wlan_hostwake  = <&pio PG 10 GPIO_ACTIVE_HIGH>;*/
                /*wlan_power    = "VCC-3V3";*/
                /*wlan_power_vol = <3300000>;*/
                /*interrupt-parent = <&pio>;*/
                /*interrupts = < PG 10 IRQ_TYPE_LEVEL_HIGH>;*/
                wakeup-source;
                regulator-boot-on;
                /*regulator-always-on;*/

            };

        };
        ...
};
...
```

tina4.0 对应路径是：

```
Tina4.0/device/config/chips/xx/configs/xxx/linux-5.4/board.dts
```

属性	说明
clocks	用于配置使用主控提供的 32k 时钟
pinctrl-0	用于配置 pin 的复用功能
pinctrl-names	用于配置 pin state
wlan_busnum	表示 WiFi 所使用的 SDIO 控制器号
wlan_power	表示给 WiFi 模组供电的 regulator 名称
wlan_io_regulator	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_regon	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_hostwake	表示 WiFi 唤醒主控的 GPIO
chip_en	表示 WiFi 模组使能引脚，硬件未使用时不配置
power_en	表示模块外部的电源开关控制引脚

说明

以上所有项必须参看原理图进行配置，配置与原理图实际使用的资源保持一致；最好是和硬件同事一起确认，比如有些设计供电可能是没有 axp 的，硬件直接供电了，所以不需要配置。

3.4.4.3 方案 module 适配

在 tina5.0/openwrt/target/h135/h135-p1/modules.mk 中添加模块配置。

```
define KernelPackage/net-aic-usb-sdio
 SUBMENU:=$(WIRELESS_MENU)
 TITLE:=aic-usb-sdio support (staging)
 DEPENDS:= +@IPV6
 KCONFIG:=\
    CONFIG_AIC8800_BTLPM_SUPPORT=m \
    CONFIG_AIC8800_WLAN_SUPPORT=m \
    CONFIG_AIC_USB_SDIO_WLAN_SUPPORT=m \
    CONFIG_PM=y\
    CONFIG_RFKILL=y \
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    CONFIG_RFKILL_PM=y \
    CONFIG_RFKILL_GPIO=y

 FILES+=$(LICHEE_OUT_DIR)/$(LICHEE_IC)/kernel/build/bsp/drivers/net/wireless/aic_usb_sdio/aic8800_bsp/
       aic8800_bsp.ko
 FILES+=$(LICHEE_OUT_DIR)/$(LICHEE_IC)/kernel/build/bsp/drivers/net/wireless/aic_usb_sdio/aic8800_fdrv/
       aic8800_fdrv.ko
 AUTOLOAD:=$(call AutoProbe, aic8800_bsp aic8800_fdrv)
endef

define KernelPackage/net-aic-usb-sdio/description
 Kernel modules for aic-usb-sdio support
endef

$(eval $(call KernelPackage,net-aic-usb-sdio))
```

tina4.0 对应路径是：

```
tina4.0/target/allwinner/xxxx/modules.mk
```

3.4.4.4 添加 Firmware

1. 添加对应 firmware 文件

在 tina5.0/platform/allwinner/wireless/firmware/aic8800 下执行：

```
mkdir usb/aic8800d80 -p
```

最后将模组厂提供的 firmware 文件放在以下路径即可。

```
tina5.0/platform/allwinner/wireless/firmware/aic8800/usb/aic8800d80
 ├──── fmacfw_8800d80_u02.bin
 ├──── fw_adid_8800d80_u02.bin
 ├──── fw_patch_8800d80_u02.bin
 ├──── fw_patch_table_8800d80_u02.bin
 └──── lmacfw_rf_8800d80_u02.bin
```

2. 修改 aic8800.mk 文件

修改 tina5.0/openwrt/openwrt/package/firmware/linux-firmware/aic8800.mk 文件

```
Package/aic8800-firmware = $(call Package/firmware-default,AIC aic8800 firmware)

SRC_CODE_DIR:=$(AIC8800_FW)

define Package/aic8800-firmware/install
    $(INSTALL_DIR) $(1)/$(FIRMWARE_PATH)
ifeq ($(CONFIG_AIC8800_USE_USB_SDD), y)
    $(INSTALL_DIR) $(1)/$(FIRMWARE_PATH)aic8800d80/

    $(INSTALL_DATA) $(SRC_CODE_DIR)/usb/aic8800d80/fmacfw_8800d80_u02.bin $(1)/$(FIRMWARE_PATH)/
      aic8800d80
    $(INSTALL_DATA) $(SRC_CODE_DIR)/usb/aic8800d80/fw_adid_8800d80_u02.bin $(1)/$(FIRMWARE_PATH)/
      aic8800d80
    $(INSTALL_DATA) $(SRC_CODE_DIR)/usb/aic8800d80/fw_patch_8800d80_u02.bin $(1)/$(FIRMWARE_PATH)/
      aic8800d80
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    $(INSTALL_DATA) $(SRC_CODE_DIR)/usb/aic8800d80/fw_patch_table_8800d80_u02.bin $(1)/$(FIRMWARE_PATH)/
      aic8800d80
ifeq ($(CONFIG_PACKAGE_aic8800-rftest), y)
    $(INSTALL_DATA) $(SRC_CODE_DIR)/usb/aic8800d80/lmacfw_rf_8800d80_u02.bin $(1)/$(FIRMWARE_PATH)/
      aic8800d80
endif
else
    $(INSTALL_DIR) $(1)/$(FIRMWARE_PATH)aic8800d80/

    $(INSTALL_DATA) $(SRC_CODE_DIR)/*.* $(1)/$(FIRMWARE_PATH)
    $(INSTALL_DATA) $(SRC_CODE_DIR)/sdio/aic8800d80/*.* $(1)/$(FIRMWARE_PATH)aic8800d80
endif
endef
$(eval $(call BuildPackage,aic8800-firmware))
```

aic8800.mk 中只是对 firmware 文件做了简单的拷贝动作，是在原先 SDIO 版本上兼容了 USB 版本的 firmware 文件拷贝。

说明

firmware 文件在编译后一般在系统 lib/firmware/。AIC 驱动使用 request_firmware 函数在内核定义的 firmware 路径中查找对应名字，一般是在 lib/firmware/下查找，驱动版本不一致 PATH 的定义的位置和名称可能存在差异，按照驱动定义的实际 PATH 位置、名称修改路径。

接着在任意目录执行 m menuconfig 就可以看到新添加的 firmware 配置。

```
> Firmware
<*> aic8800-firmware..... AIC aic8800 firmware
[*] aic8800 with usb sdd
```

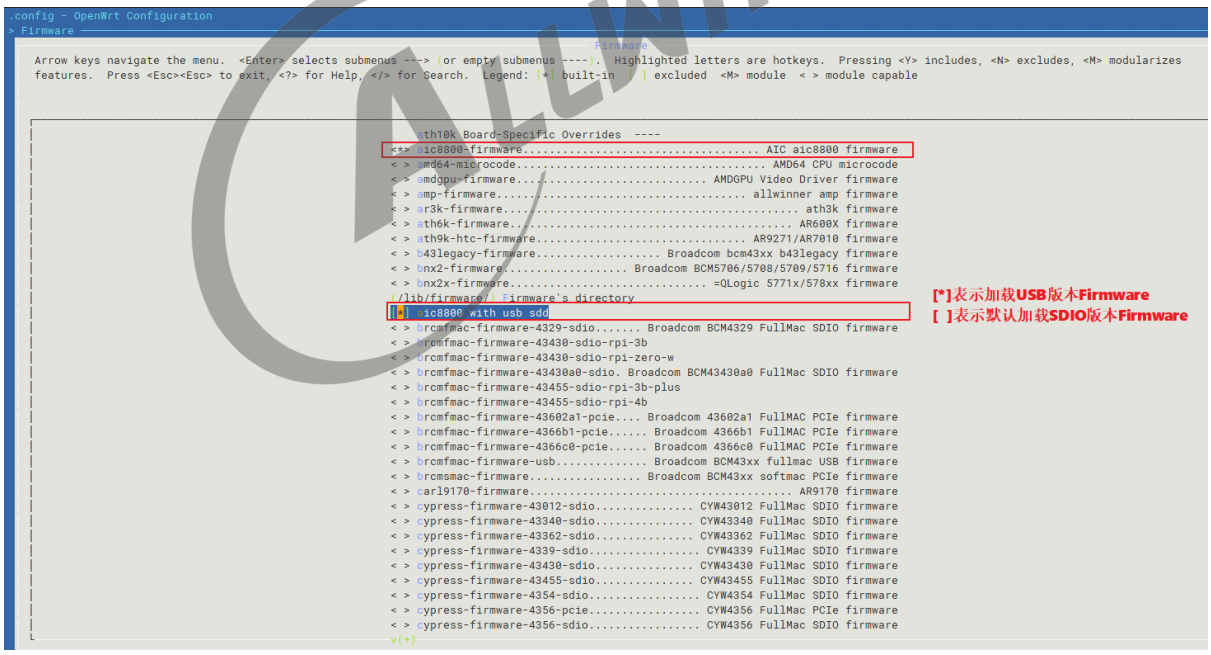


图 3-80: Tina5.0-aic_usb_sdio-firmware 配置

3.4.4.5 应用工具适配

1.wifimanager 配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

m menuconfig

> Allwinner > Wireless
<*> wifimanager-v2.0..... Tina wifimanager-v2.0 --->

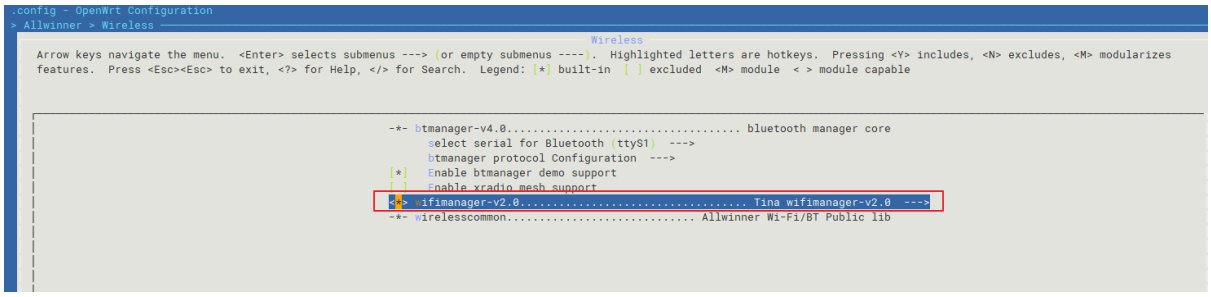


图 3-81: Tina5.0-wifimanager 配置

2.wifimanager demo (DEMO_TEST_TOOL)

这就是原先 demo 的选项，现在提供更多 demo，所以做了区分，原先添加 wifi、wifi_damon 程序的 demo 是 DEMO_TEST_TOOL。

m menuconfig

> Allwinner > Wireless > wifimanager-v2.0
<*> wifimanager-v2.0-demo..... Tina wifimanager-v2.0 app demo
Wifimanager demo choice (DEMO_TEST_TOOL) --->

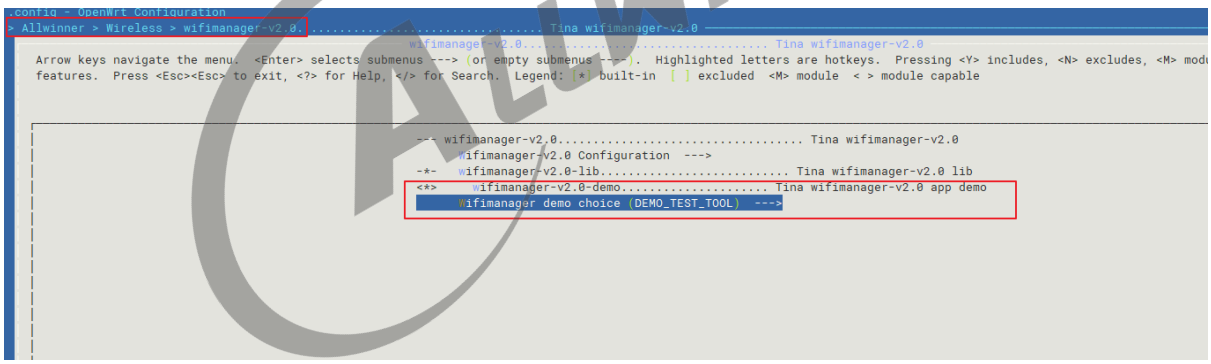


图 3-82: Tina5.0-wifimanager demo 配置

3. 配网配置

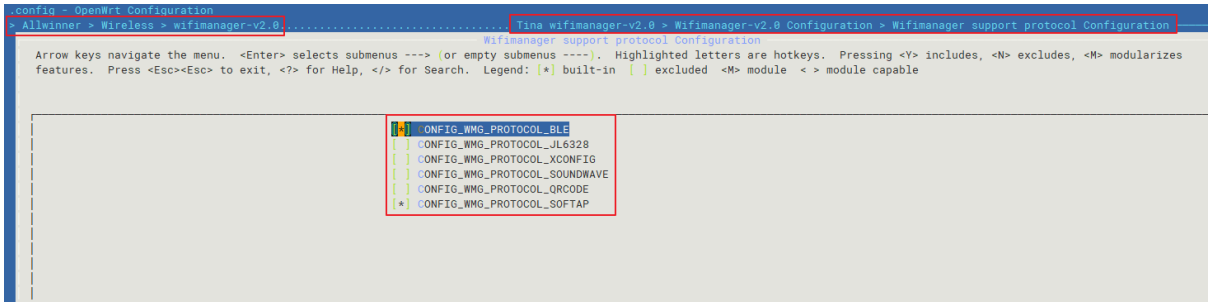


图 3-83: Tina5.0-wifimanager 配网方式配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

其中各项配网分别是：

- CONFIG_WMG_PROTOCOL_BLE：蓝牙配网
- CONFIG_WMG_PROTOCOL_JL6328：杰理芯片 JL6328 蓝牙配网
- CONFIG_WMG_PROTOCOL_XCONFIG：XCONFIG 配网
- CONFIG_WMG_PROTOCOL_SOUNDWAVE：声波配网
- CONFIG_WMG_PROTOCOL_QRCODE：摄像头二维码配网
- CONFIG_WMG_PROTOCOL_SOFTAP：softap 配网

4.iperf 工具配置

```
m menuconfig

> Network
```

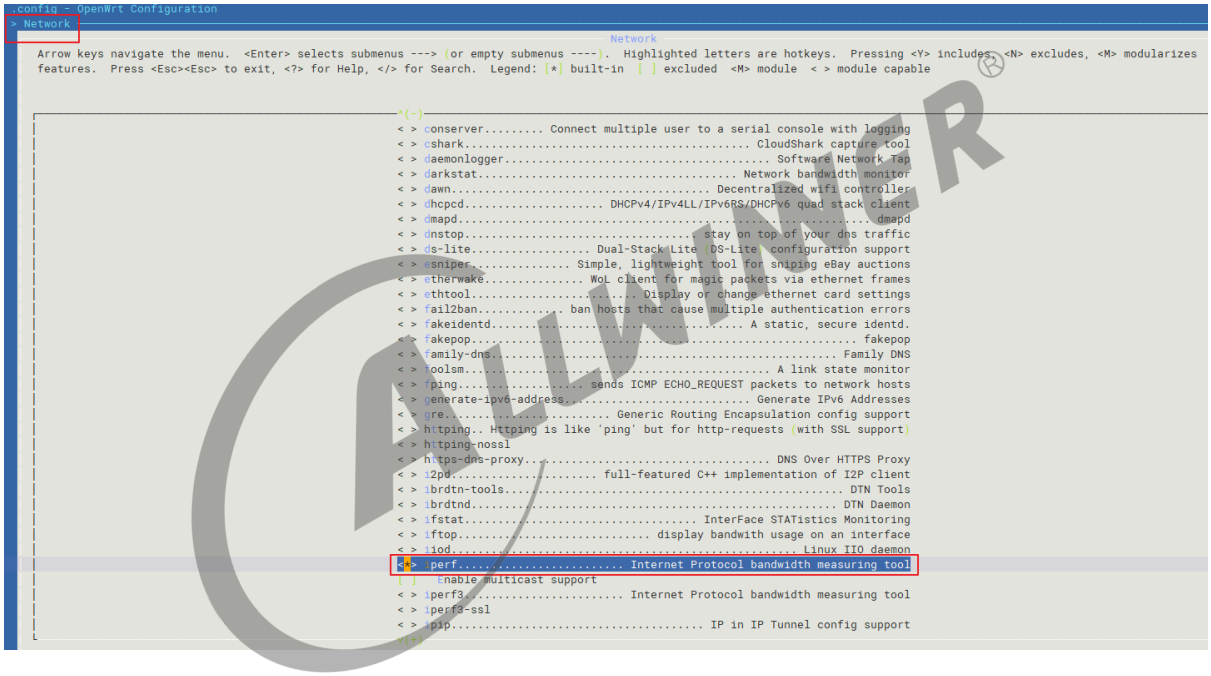


图 3-84: Tina5-iperf 配置

5.wpad 配置

wpad 工具包括 wpa-supplicant 服务和 hostapd 服务。

```
m menuconfig

> Network > WirelessAPD
<*> wpad..... IEEE 802.1x Auth/Supplicant (built-in full)
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

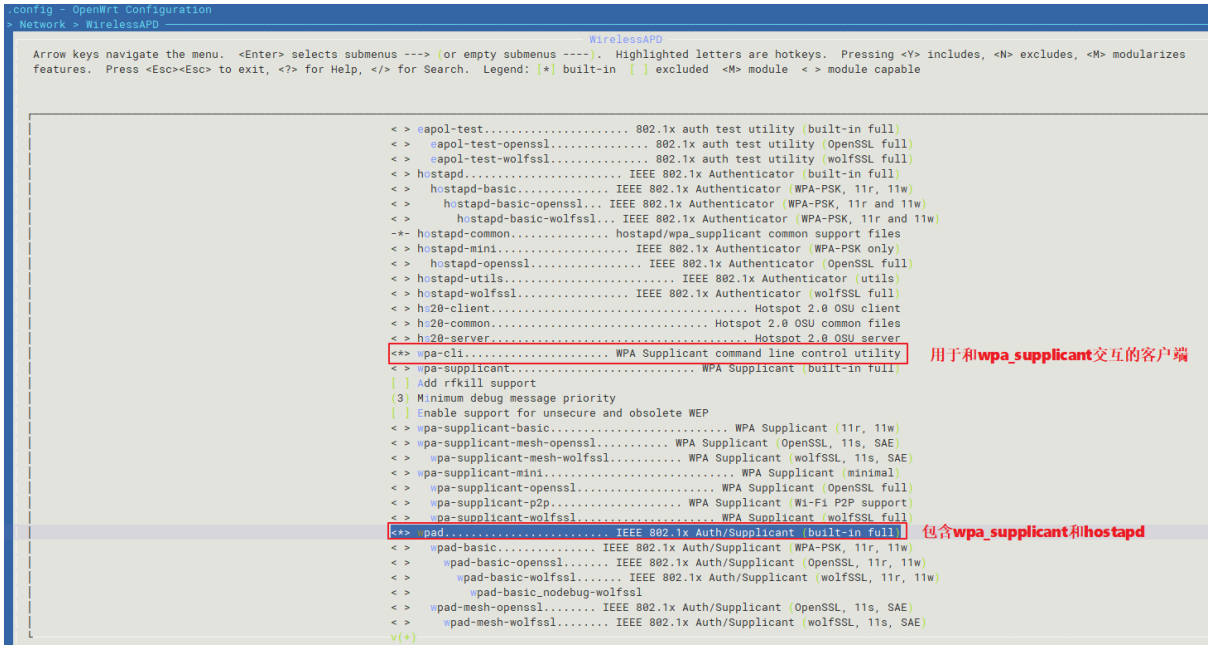


图 3-85: Tina5.0-wpad 配置

3.4.5 AIC8800D(buildroot 编译系统)

主控：T527

无线模组：AW869A

内核版本：linux-5.15

方案：demo_linux_car

3.4.5.1 内核驱动适配

内核驱动适配分为如下四个步骤：

- 获取源码。
- 添加 Kconfig 和 Makefile 配置。
- make kernel_menuconfig 配置。
- 编译。

1. 获取驱动源码，放到内核驱动路径下。客户可以从一号通账号获取。

```
Tina5.0/bsp/drivers/net/wireless/aic8800 ├── aic8800_bsp ├── aic8800_btlpm ├── aic8800_fdrv ├── Kconfig
└── Makefile
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

2. 内核添加 Kconfig 和 Makefile 的配置。

tina-5.0/bsp/drivers/net/wireless/Kconfig 文件中引入 AIC8800D 驱动的 Kconfig 配置。

source "drivers/net/wireless/aic8800/Kconfig"

tina-5.0/bsp/drivers/net/wireless/Makefile 文件中引入 AIC8800D 驱动的 Makefile 配置。

obj-$(CONFIG_AIC_WLAN_SUPPORT) += aic8800/

3../build.sh menuconfig 配置

SDK 根内核目录执行./build.sh menuconfig。

a.AIC8800D 驱动的配置

> Allwinner BSP > Device Drivers > Network Device Drivers > Wireless LAN [*] AIC wireless Support Enable Chip Interface (SDIO interface support) <M> AIC8800 wlan Support <M> AIC8800 bluetooth Support (UART)

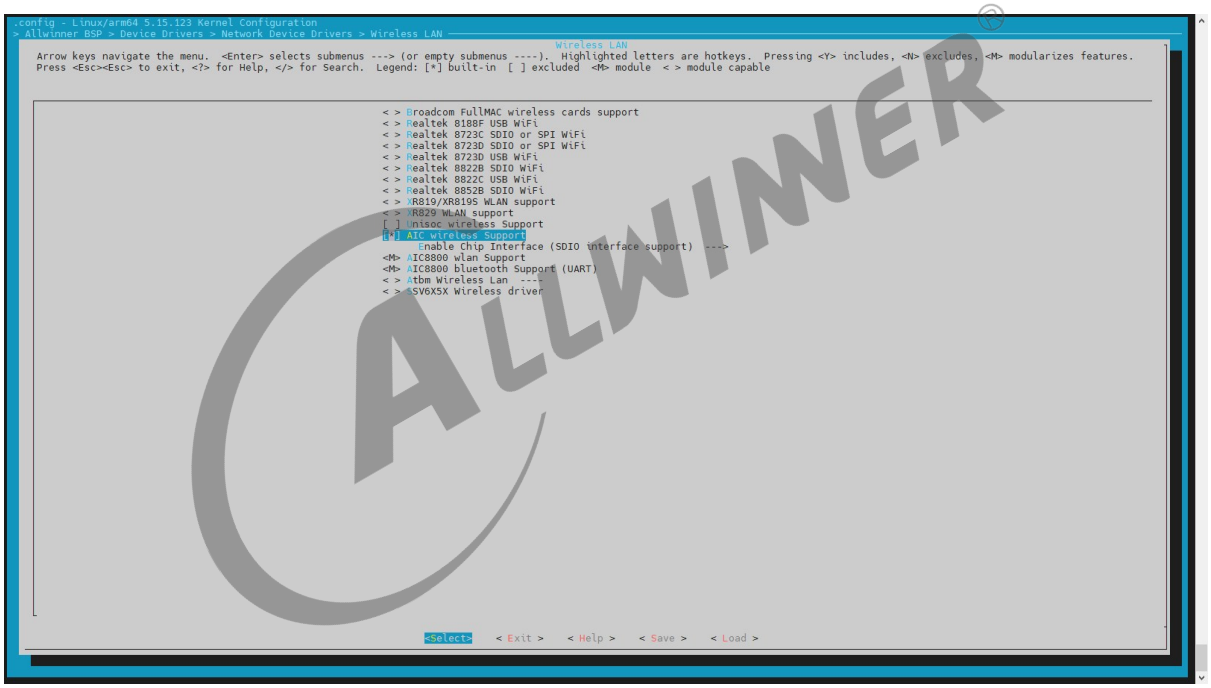


图 3-86: buildroot-aic8800D 内核配置

b. sunxi-rf 的配置

> Allwinner BSP > Device Drivers > Misc devices
  <*> Allwinner rfkill driver

版权所有 © 珠海全志科技股份有限公司。保留一切权利

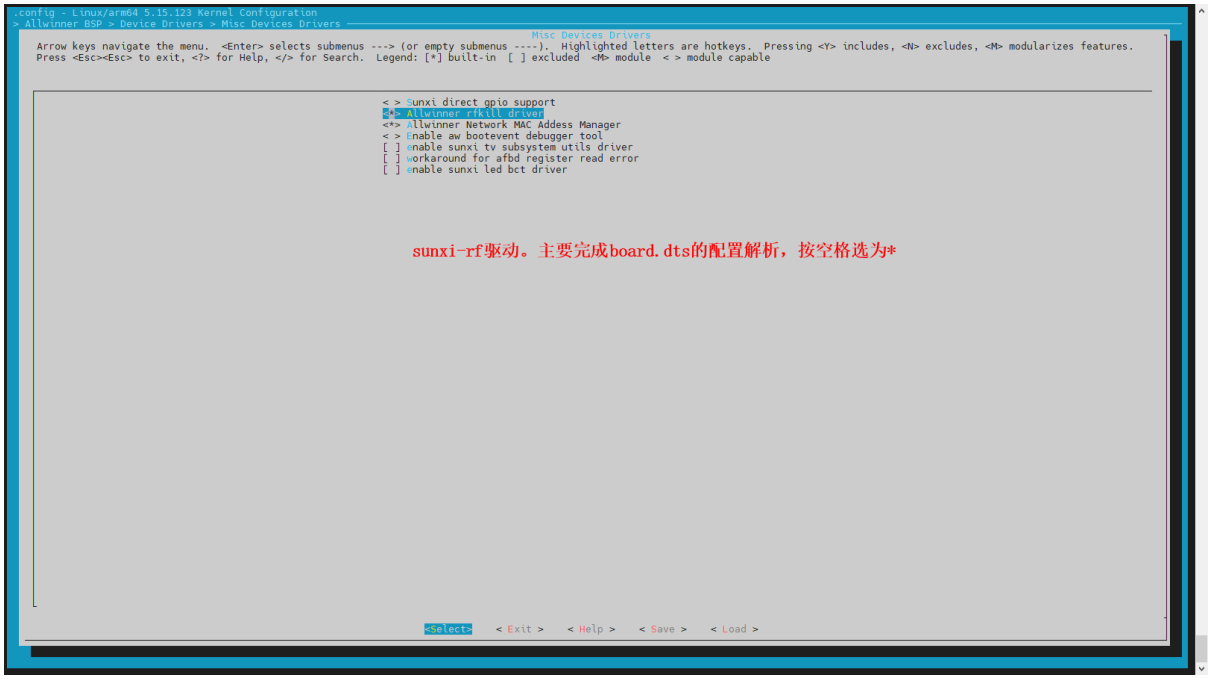


图 3-87: buildroot-sunxi-rf 配置

c. SDIO 的配置

> Allwinner BSP > Device Drivers > SD/MMC Drivers
<*> Allwinner SD/MMC Host Controller Support

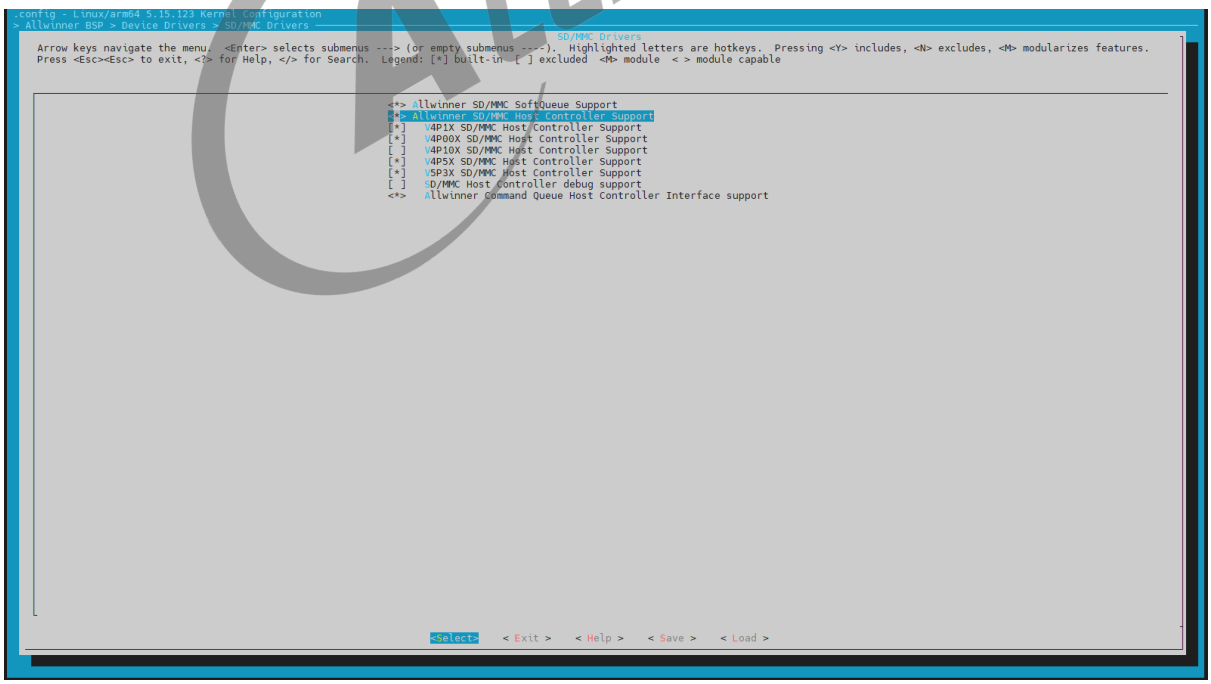


图 3-88: buildroot-sdc 配置

4. 编译

版权所有 © 珠海全志科技股份有限公司。保留一切权利

在 tina-5.0 目录执行./build.sh kernel 进行编译。

```
tina-5.0/out/t527/kernel/build/bsp/drivers/net/wireless/aic8800
.
├── aic8800_bsp
│   ├── aic8800_bsp.ko
├── aic8800_btlpm
│   ├── aic8800_btlpm.ko
├── aic8800_fdrv
│   ├── aic8800_fdrv.ko
```

3.4.5.2 硬件资源适配

在 tina-5.0/device/config/chips/t527/configs/demo_linux_car 中添加引脚配置。

```
&pio {
    vcc-pg-supply = <&reg_pio1_8>;  /*AIC8800的IO电压为1.8V，这里需要配置PG口1.8V*/
    sdc0_pins_a: sdc0@0 {
        ...
    };
...
&rfkill {
  compatible = "allwinner,sunxi-rfkill";
  chip_en;
  power_en;
  pinctrl-0;
  pinctrl-names;
  status = "okay";

  /* wlan session */
  wlan {
    compatible   = "allwinner,sunxi-wlan";
    clocks;
    clock-names;
    wlan_power   = "axp2202-aldo3", "axp2202-bldo1"; /* vcc-pl/vcc-pg/vcc-pm */
    wlan_power_vol= <3300000>, <1800000>;
    wlan_busnum  = <0x1>;
    wlan_regon   = <&r_pio PM 1 GPIO_ACTIVE_HIGH>;
    wlan_hostwake = <&r_pio PM 0 GPIO_ACTIVE_HIGH>;
    wakeup-source;
  };

  /* bt session */
  bt {
    compatible   = "allwinner,sunxi-bt";
    clocks;
    clock-names;
    bt_power   = "axp2202-aldo3", "axp2202-bldo1"; /* vcc-pl/vcc-pg/vcc-pm */
    bt_power_vol= <3300000>, <1800000>;
    bt_rst_n     = <&r_pio PM 2 GPIO_ACTIVE_LOW>;
    };
  };

  &addr_mgt {
    compatible    = "allwinner,sunxi-addr_mgt";
    type_addr_wifi = <0x0>;
    type_addr_bt  = <0x0>;
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
    type_addr_eth = <0x0>;
    status        = "okay";
  };

  &btlpm {
    compatible = "allwinner,sunxi-btlpm";
    uart_index = <0x1>;
    bt_wake    = <&r_pio PM 3 GPIO_ACTIVE_HIGH>;
    bt_hostwake = <&r_pio PM 4 GPIO_ACTIVE_HIGH>;
    wakeup-source;
    status     = "okay";
  };

...
```

属性	说明
wlan_busnum	表示 WiFi 所使用的 SDIO 控制器号
wlan_power	表示给 WiFi 模组供电的 regulator 名称
wlan_power_vol	表示 WiFi 模组需要使用到的电的电压
wlan_regon	表示给 WiFi 模组的 GPIO 供电的 regulator 名称
wlan_hostwake	表示 WiFi 唤醒主控的 GPIO
chip_en	表示 WiFi 模组使能引脚，硬件未使用时不配置
power_en	表示模块外部的电源开关控制引脚

说明

以上所有项必须参看原理图进行配置，配置与原理图实际使用的资源保持一致；最好是和硬件同事一起确认，比如有些设计供电可能是没有 axp 的，硬件直接供电了，所以不需要配置，特别注意的就是 sdio 的配置。

3.4.5.3 方案 module 适配

说明

buildroot 编译方式无 module 适配，系统起来后需手动加载模组或根据实际情况自编写启动自加载脚本。

3.4.5.4 添加 Firmware

在 tina-5.0/platform/allwinner/wireless/firmware/aic8800 添加 aic8800 需要的 firmware。

```
tina-5.0/platform/allwinner/wireless/firmware/aic8800
├── aic8800d80
│   ├── aic_userconfig_8800d80-Typ.txt
│   ├── fmacfw_8800d80.bin
│   ├── fmacfw_8800d80_u02.bin
│   ├── fmacfw_rf_8800d80.bin
│   ├── fmacfw_rf_8800d80_u02.bin
│   ├── fw_adid_8800d80.bin
│   ├── fw_adid_8800d80_u02.bin
│   ├── fw_patch_8800d80.bin
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
│   ├── fw_patch_8800d80_u02.bin
│   ├── fw_patch_table_8800d80.bin
│   ├── fw_patch_table_8800d80_u02.bin
│   ├── lmacfw_rf_8800d80.bin
│   └── lmacfw_rf_8800d80_u02.bin
├── aic_userconfig.txt
├── fmacfw.bin
├── fmacfw_rf.bin
├── fmacfw_rf_usb.bin
├── fmacfw_usb.bin
├── fw_adid.bin
├── fw_adid_u03.bin
├── fw_patch.bin
├── fw_patch_table.bin
├── fw_patch_table_u03.bin
└── fw_patch_u03.bin
```

修改 tina-5.0/buildroot/config/buildroot/wifi-firmware/wifi-firmware.mk 文件

```
define WIFI_FIRMWARE_INSTALL_TARGET_CMDS
  $(INSTALL) -d -m 0755 $(FIRMWARE_DIR)/
...
  if test "$(BR2_PACKAGE_AIC8800_FIRMWARE)" = "y" ; then \
  cp -r $(WIFI_FIRMWARE_SITE)/aic8800/* $(FIRMWARE_DIR) ; \
  fi
...
endef

$(eval $(generic-package))
```

说明：可以看到整个 Makefile 文件就只是做了简单的拷贝动作。

说明

一般 firmware 的路径是：系统的/lib/firmware/, 如果更改请确保是否和驱动中定义的保持一致，最新版驱动已经自适应寻找路径了, 早期的驱动版本一定要留意。

3.4.5.5 应用工具适配

1.wifimanager 配置 (buildroot 编译方式暂时不支持 wifimanager 定制化配置)

```
./build.sh buildroot_menuconfig
> Target packages > allwinner platform private package select > wireless
```

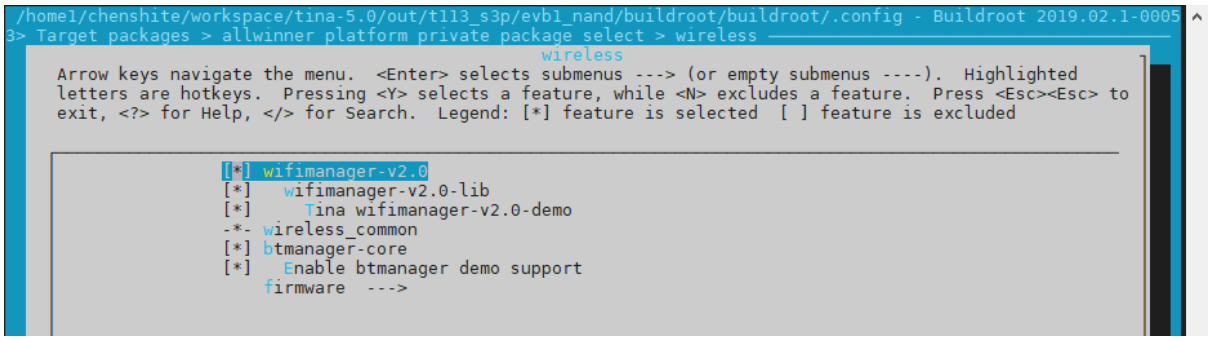


图 3-89: buildroot-wifimanager 配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

2.rf 工具配置 (buildroot 编译方式暂不支持 rf 工具)

3.iperf 工具配置

```
./build.sh buildroot_menuconfig
> Target packages > allwinner platform private package select > wireless
```

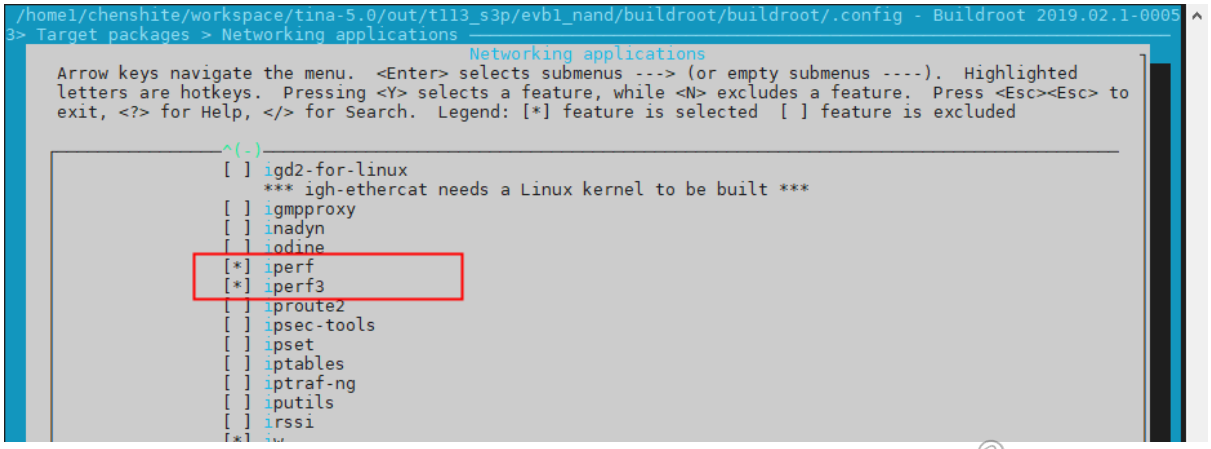


图 3-90: buildroot-iperf 工具配置

4.wpa/hostapd 工具配置

wpa 工具包括 wpa-supplicant 服务的和 wpa-cli 客户端。

```
./build.sh buildroot_menuconfig
> Target packages > Networking applications
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

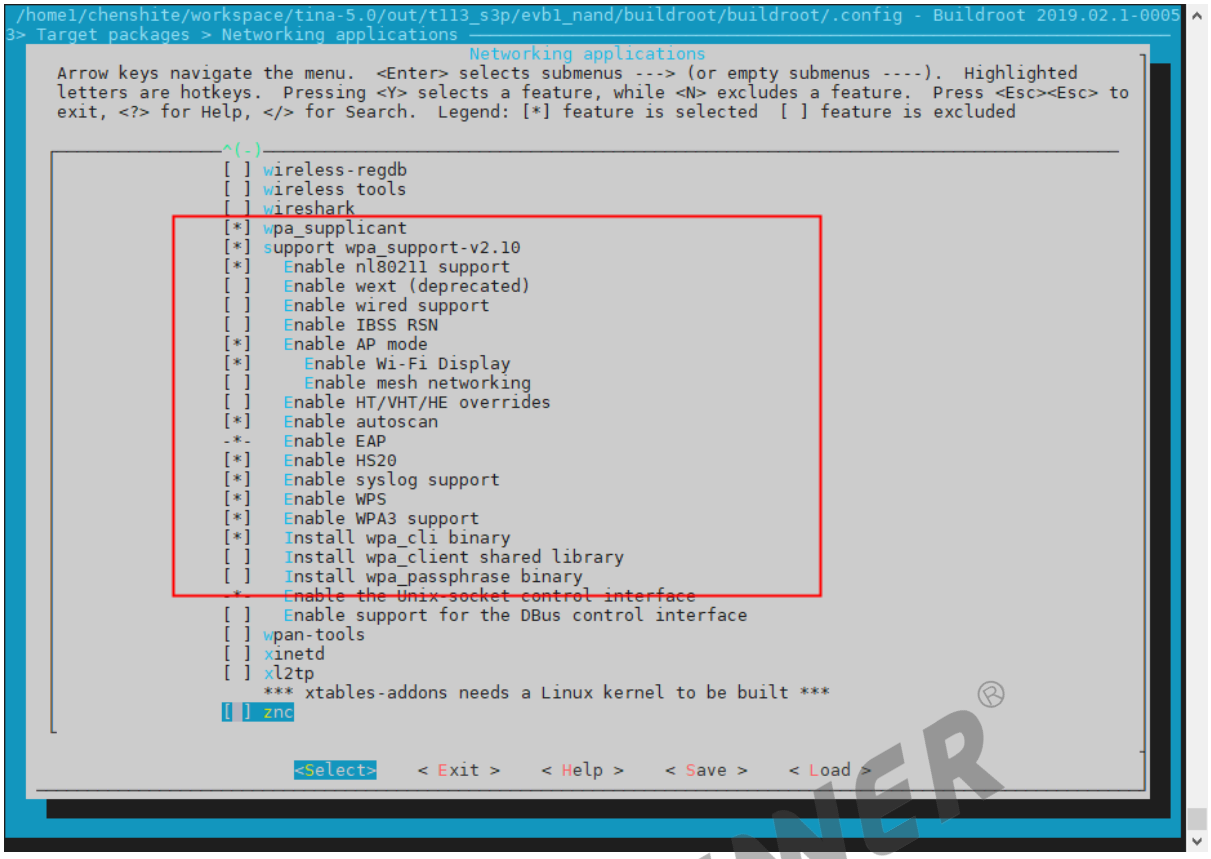


图 3-91: buildroot-wpasupplicant 配置

hostapd 工具

```
./build.sh buildroot_menuconfig
> Target packages > Networking applications
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

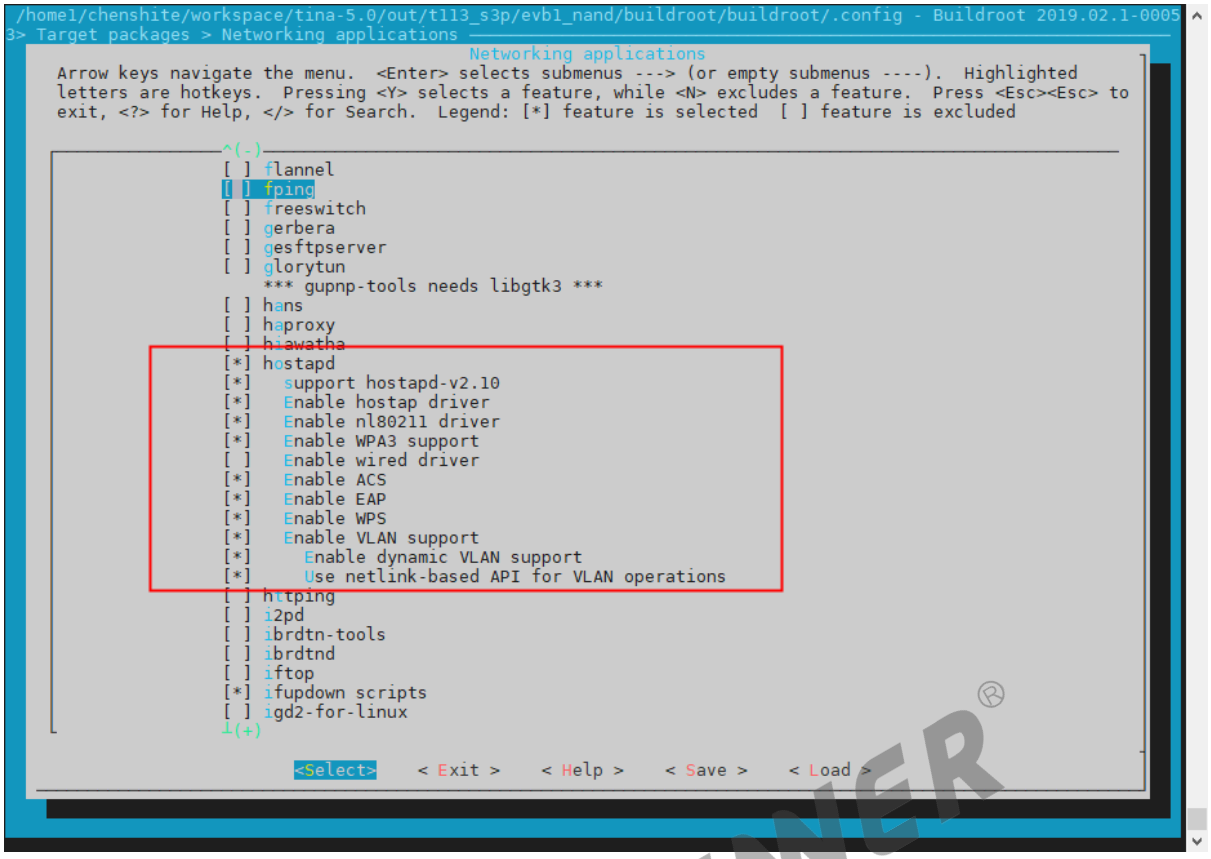


图 3-92: buildroot-hostapd 配置

版权所有 © 珠海全志科技股份有限公司。保留一切权利

4 验证说明

wifi 的基础功能可以通过下面的步骤进行验证:

- 加载 wifi 驱动，启动 wpa_supplicant
- 扫描测试
- 联网测试
- 查看 ip 地址
- ping 百度测试

说明

乐鑫 esp32 模组不通过 wpa_supplicant 来进行设置和交互，它提供了自己的配置工具，因此这里部分步骤不适用，请参考 esp32 模组章节。

4.1 加载 wifi 驱动，启动 wpa_supplicant

1). 系统上电后使用 insmod 命令加载驱动，使用 lsmod 查看驱动是否已经加载成功。部分系统已经使用自动化脚本自动加载不需手动加载。

2). 确保 wpa_supplicant 是否起来，使用 ps 命令查看 wpa_supplicant 是否起来，大部分系统由自动化脚本启动，没有启动的可以手动启动 (esp32 模组比较特殊不通过 wpa_supplicant 来进行交互)。

3). 使用 wifimanager 进行测试，Tina 系统提供了一套测试 wifi 用的测试 demo(wifimanager)，wifimanager 是封装了 wpa_cli 的一套接口但使用更简单。用户可以使用这套测试 demo 来进行测试，下面只介绍 2 个 demo 其他 demo 可以参考《Tina_Linux_Wi-Fi_ 软件开发指南》。

4.2 扫描测试

使用 wifi -s 用于扫描 AP，可以看到如下打印。

```
wifi -s
WIF: bss[00]: bssid=50:d2:f5:f1:b7:08 ssid=allwinner-wpa2 channel=6(freq=2437) rssi=124 sec=WPA2_PSK
WIF: bss[01]: bssid=f4:ec:38:70:67:a6 ssid=allwinner-wep channel=4(freq=2427) rssi=76 sec=WEP
WIF: bss[02]: bssid=88:c3:97:b1:92:76 ssid=AW-PDC-PD2-XIAOMI_9275-74 channel=1(freq=2412) rssi=82 sec=WPA_PSK
WIF: bss[03]: bssid=0c:72:2c:6d:d5:52 ssid=123 channel=7(freq=2442) rssi=84 sec=WPA_PSK
.....
WIF: ==Wi-Fi scan successful, total 19 ap(buff size: 20) time 0.000000 ms==
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

4.3 联网测试

使用命令 wifi -c ssid passwd 命令可以进行联网测试。

连接成功会有 connect success 的 log 出现。例：

```
WMG_INFO [wifi_daemon.c:cmd_handle_c:778]:  ===Wi-Fi connect successful,time 7510.000000 ms===
```

4.4 查看 ip 地址

使用 ifconfig 命令可以查看到获取到的 ip 值，该命令是标准命令，更详细说明请百度。

```
root@TinaLinux:/# ifconfig wlan0
wlan0     Link encap:Ethernet  HWaddr F4:68:5F:9A:75:1F
          inet addr:192.168.51.178  Bcast:192.168.51.255  Mask:255.255.255.0
          inet6 addr: fe80::f668:5fff:fe9a:751f/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:352 errors:0 dropped:0 overruns:0 frame:0
          TX packets:387 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:26897 (26.2 KiB)  TX bytes:24252 (23.6 KiB)
```

4.5 ping 百度测试

即使获取到了 ip 地址也不能说明基本功能已通，一般使用 ping 命令去进行测试 (前提 AP 是能连外网的)。

ping 百度测试：ping www.baidu.com。

如果出现下面类似的 log，说明网络基本功能已通。

```
PING www.a.shifen.com (36.152.44.95) 56(84) bytes of data.
64 bytes from 36.152.44.95 (36.152.44.95): icmp_seq=1 ttl=56 time=10.5 ms
64 bytes from 36.152.44.95 (36.152.44.95): icmp_seq=2 ttl=56 time=11.8 ms
64 bytes from 36.152.44.95 (36.152.44.95): icmp_seq=3 ttl=56 time=10.7 ms
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

5 排查说明

5.1 排查步骤

5.1.1 硬件工作条件排查

- 供电：先和硬件工程师确定当前模组的供电方案，用万用表或者示波器实测模组的供电。
- 使能：根据模组规格书确定 WL-REG-ON 引脚的时序规则，建议用示波器直接抓取整个驱动加载过程 WL-REG-ON 的时序波形图。若 WL-REG-ON 未启动，先排查 WL-REG-ON 不工作原因，WL-REG-ON 不按规定时序工作则 WiFi 模组就无法工作。
- 时钟：直接用示波器抓取 SDC_CLK，24M，32K 的信号波形图。
- 外围：硬件同事审核外围器件的焊接，如电阻电容贴片，天线通路，以及模组是否存在虚焊。

5.1.2 软件配置排查

软件配置的排查相对比较多，请按照 2.3 节逐一排查相关文件的修改，这里就不再重复展开。

说明

软件配置的排查请在确认上述硬件工作条件都 OK 后进行。

5.1.3 软件流程定位排查

说明

软件流程的排查请在确认上述硬件工作条件都 OK 后进行。

(1) 系统启动：系统启动时由 sunxi-rf 驱动负责配置的解析，在内核启动阶段一般可以看到如下类似打印：

```
[   1.117385] sunxi-rfkill soc@3000000:rfkill@0: module version: v1.0.9
[   1.139386] sunxi-rfkill soc@3000000:rfkill@0: wlan_busnum (1)
[   1.145919] sunxi-rfkill soc@3000000:rfkill@0: Missing wlan_power.
[   1.152844] sunxi-rfkill soc@3000000:rfkill@0: wlan clock[0] (32k-fanout1)
[   1.160554] sunxi-rfkill soc@3000000:rfkill@0: wlan_regon gpio=204 assert=1
[   1.168408] sunxi-rfkill soc@3000000:rfkill@0: wlan_hostwake gpio=202 assert=1
[   1.176569] sunxi-rfkill soc@3000000:rfkill@0: wakeup source is enabled
```

(2) 驱动加载：驱动加载整体过程相对比较复杂，但是我们可以细化到如下几个过程。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

【命令：insmod /…/…/xxx.ko】具体模组 ko 的路径。

a. 上电：该过程主要还是在 sunxi-rf 驱动完成的，主要接口【sunxi_wlan_on()】。一般可以看到如下类似打印。

```
[ 351.636987] sunxi-rfkill soc@3000000:rfkill@0: bus_index: 1
[ 351.662971] sunxi-rfkill soc@3000000:rfkill@0: wlan power on success
```

b. 扫卡：该过程主要是 sdio 驱动的操作，主要接口【sunxi_mmc_rescan_card()】。一般可以看到如下类似打印。

```
[ 351.891832] sunxi-mmc 4021000.sdmmc: no vqmmc,Check if there is regulator
[ 351.916462] sunxi-mmc 4021000.sdmmc: sdc set ios:clk 400000Hz bm PP pm ON vdd 21 width 1 timing LEGACY(SDR12)
      dt B
[ 351.928252] sunxi-mmc 4021000.sdmmc: failed to get DS26_SDR12 used default
[ 351.949071] sunxi-mmc 4021000.sdmmc: sdc set ios:clk 400000Hz bm PP pm ON vdd 21 width 1 timing LEGACY(SDR12)
      dt B
[ 351.960857] sunxi-mmc 4021000.sdmmc: failed to get DS26_SDR12 used default
[ 351.971581] sunxi-mmc 4021000.sdmmc: sdc set ios:clk 400000Hz bm PP pm ON vdd 21 width 1 timing LEGACY(SDR12)
      dt B
[ 351.983339] sunxi-mmc 4021000.sdmmc: failed to get DS26_SDR12 used default
[ 352.001647] sunxi-mmc 4021000.sdmmc: sdc set ios:clk 400000Hz bm PP pm ON vdd 21 width 1 timing SD-HS(SDR25)
      dt B
[ 352.013317] sunxi-mmc 4021000.sdmmc: failed to get HSSDR52_SDR25 used default
[ 352.021320] sunxi-mmc 4021000.sdmmc: sdc set ios:clk 50000000Hz bm PP pm ON vdd 21 width 1 timing SD-HS(
      SDR25) dt B
[ 352.033225] sunxi-mmc 4021000.sdmmc: failed to get HSSDR52_SDR25 used default
[ 352.041281] sunxi-mmc 4021000.sdmmc: sdc set ios:clk 50000000Hz bm PP pm ON vdd 21 width 4 timing SD-HS(
      SDR25) dt B
[ 352.053173] sunxi-mmc 4021000.sdmmc: failed to get HSSDR52_SDR25 used default
[ 352.062120] mmc1: new high speed SDIO card at address 0001   【成功扫到卡】
```

c. 下载 Firmware：该过程并不是所有厂商模组都会有的动作，主要接口都是具体厂商驱动实现的，就不列举了。一般可以看到如下类似打印。以 XRADIO 为例：

```
[ 352.171587] [XRADIO] Bootloader complete
[ 352.262351] [XRADIO] Firmware completed.   【下载firmware成功】
[ 352.268727] [WSM] Firmware Label:XR_C09.08.52.73_DBG_02.122 2GHZ HT40 May 18 2021 13:36:09
[ 352.286326] [XRADIO] Firmware Startup Done.
```

说明

通过 lsmod 查看对应驱动是否已经成功加载。

(3) 网络服务启动：wpa_supplicant。

可以通过如下命令 ps 确认

```
root@TinaLinux:/# ps | grep wpa_supplicant
 1268 root     2620 S    grep wpa_supplicant
```

如果没有默认启动，可以通过如下命令打开：

版权所有 © 珠海全志科技股份有限公司。保留一切权利

```
./etc/init.d/wpa_supplicant start
```

(4) 启动网卡：ifconfig wlan0 up

至此基本就可以正常使用 Wi-Fi 了，可以用 wifimanager 工具进行扫描测试。

wifi -s

确保供电正常的前提下，排查思路操作：

1. 系统上电进入控制台执行 lsmod 查看相关模组（即 xradio_wlan，xradio_core，xradio_mac）是否已经加载成功。

```
root@TinaLinux:/# lsmod
...
xradio_core        529408  1 xradio_wlan
xradio_mac         253247  1 xradio_core
xradio_wlan          1050  0
```

2. 若加载失败，优先分析启动 log（如果没有启动 log，则手动加载 ko）。执行命令如下：

```
root@TinaLinux:/# insmod /lib/modules/4.9.191/xradio_mac.ko
root@TinaLinux:/# insmod /lib/modules/4.9.191/xradio_core.ko
root@TinaLinux:/# insmod /lib/modules/4.9.191/xradio_wlan.ko
```

寻找关键字：确立正常扫卡

```
[XRADIO] Detect SDIO card 1
...
mmc1: new high speed SDIO card at address 0001
[SBUS] XRadio Device:sdio clk=50000000
```

若扫卡失败分析方案 board.dts

```
wlan: wlan@0 {
        ...
        wlan_busnum    = <0x1>;
}
```

3. 若加载失败，分析启动 log（如果没有启动 log，则手动加载 ko）。

寻找关键字：确立 Firmware 正常下载。

```
[XRADIO] XRADIO_HW_REV 1.0 detected.
[XRADIO] Bootloader complete
[XRADIO] Firmware completed.
[WSM] Firmware Label:XR_C09.08.52.64_DBG_02.100 2GHZ HT40 Jan  3 2020 13:14:37
[XRADIO] Firmware Startup Done.
```

若下载 firmware 失败, 分析版本是否和模组匹配，晶振是 24M 还是 40M 的，是否版本太旧不兼容以及 firmware 的路径。

如：Write hwinfo data to file:/data/vendor/wifi/hwinfo.bin failed!

表示 firmware 驱动的路径和 Tina 系统的不一致。

版权所有 © 珠海全志科技股份有限公司。保留一切权利

4. 若加载失败，分析启动 log(如果没有启动 log，则手动加载 ko)。

寻找关键字：确立是否有异常。

```
[XRADIO_ERR] xradio_request_gpio_irq: request_irq err: -22
[SBUS_ERR] sdio_irq_subscribe:xradio_request_gpio_irq failed(-1).
```

以上打印表示中断的 gpio 配置错误，分析 board.dts。

```
wlan: wlan@0 {
        ...
        wlan_hostwake = <&r_pio PL 6 6 0x1 0x2 0>;
}
```

```
[XRADIO_ERR] can't open /etc/wifi/xr_wifi.conf, failed(-30)
[XRADIO_ERR] Access_file failed, path:/etc/wifi/xr_wifi.conf!
```

以上打印表示找不到文件，该文件是用来保存 mac 地址的。

5.2 常见问题

按照以上步骤排查完并确立都正常后，Wi-Fi 仍然有问题，请建 Aservice 或者 PMS 导入。

问题背景：客户（内部方案）+ 系统版本 + 硬件主控 + 硬件 Wi-Fi devices。

问题描述：一句话概况问题。【（偶现，低概率，高概率，必现）】

问题分析：

1. 确立当前驱动，FW，ETF 工具，ETF 固件版本乃至系统版本（可根据实际情况调整）。

驱动版本：查看启动 log。

```
======== XRADIO WIFI OPEN ========
[XRADIO] Driver Label:XR_V02.16.83 _HT40_01.33  Oct 20 2020 12:47:22
```

FW 版本：查看启动 log。

```
[SBUS] XRadio Device:sdio clk=50000000
[XRADIO] XRADIO_HW_REV 1.0 detected.
[XRADIO] Bootloader complete
[XRADIO] Firmware completed.
[WSM] Firmware Label:XR_C09.08.52.64_DBG_02.100 2GHZ HT40 Jan  3 2020 13:14:37
```

ETF 工具版本：执行 etf 命令。

```
etf --version
ETF version: 1.3.6
```

etf 固件版本：执行 etf 连接命令。

```
etf connect
[WSM] Firmware Label:ETF_FW_A09.01.0102-HIF; Mar  6 2020, 11:29:47
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

Tina 系统版本：

```
cat /etc/openwrt_release
DISTRIB_DESCRIPTION='tina.xxx.20201208.012018 3.5.1'
```

2. 已经做过如上排查，如：供电是否正常，regon 是否上拉，sdio_clk 是否正常起振，驱动是否正常加载，wlan0 是否正常起卡，wpa_supplicant 服务是否启动。

问题环境：

1. 直接提供可用的问题环境。

2. 详细告知如何搭建问题环境。

问题复现：

准确列出复现步骤。

附上已做测试的 Log。

其他注意：

建 PMS 的同时明确优先级和紧急程度。

例如：

```
问题背景：内部方案+Tina+R329+XR829。

问题描述：水星路由器办公环境测试吞吐无法和样机相互ping通。【必现】

问题分析：
1.版本信息
Tina版本：DISTRIB_DESCRIPTION='tina.xxx.20201208.012018 3.5.1'
XR829驱动版本：Driver Label:XR_V02.16.83 _HT40_01.33  Oct 20 2020 12:47:22
Firmware版本：Firmware Label:XR_C09.08.52.64_DBG_02.100 2GHZ HT40 Jan  3 2020 13:14:37
2.初步分析
2.1替换rtl8723ds测试问题仍然从存在。---与Wi-Fi模组无关。
2.2替换小米路由器测试XR829和rtl8723ds两款模组问题不存在。---可能是路由器兼容性问题。

问题复现：
1.R329+XR829板子烧写20201208.012018 3.5.1固件。
2.系统启动后使用wifi_connect_ap_test联上水星路由器，并通过ifconfig查看ip地址。
3.pc通过网线和路由器的lan口相连，并通过cmd窗口执行ipconfig查看ip地址。
4.pc和样机相互ping ip。
```

版权所有 © 珠海全志科技股份有限公司。保留一切权利

著作权声明

版权所有 ©2024 珠海全志科技股份有限公司。保留一切权利。

本文档及内容受著作权法保护，其著作权由珠海全志科技股份有限公司（“全志”）拥有并保留一切权利。

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

商标声明

、、全志科技、（不完全列举）均为珠海全志科技股份有限公司的商标或者注册商标。在本文档描述的产品中出现的其它商标，产品名称，和服务名称，均由其各自所有人拥有。

免责声明

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

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

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