AW Tina Linux 蓝牙
模组移植指南

版本号: 1.0
发布日期: 2022-03-21

版本历史

版本号	日期	制/修订人	内容描述
1.0	2021.08.23	AW BT Team	创建

目　录

插　图

1 前言

1.1 文档简介

本文档主要介绍 Tina Linux 平台上蓝牙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 平台上适配一款蓝牙模组。

1.2 目标读者

Tina Linux 开发用户、蓝牙开发使用人员。

1.3 适用范围

Tina Linux 平台

2 简述

蓝牙模组移植分为以下两类，其移植方法不同，难度依次递增，在移植之前建议用户先检查下支持列表，判断当前需移植的物料是否在其他方案已支持，如果其他平台方案有支持，参考章节。难度比较大是新的物料（所有平台方案都未支持过），可以参考章节。本文将首先介绍下模组移植的基本原理和套路，使用户能够掌握模组移植的核心技能，以不变应万变，因此建议用户在移植模组前能够认真研读章节；其次会对两类模组移植进行示例阐述，方便用户能够快速上手理解（用户可以根据当前移植的物料选择对应章节阅读）；最后会列出模组移植常见的问题，便于提供用户遇到问题可以进行参考。

- 平台方案已支持。
- 平台方案未支持，但是其他方案有支持。

注：平台方案指的是Tina Linux的工程，Tina Linux在编译的时候会进行lunch动作，选择方案。

3 支持列表

芯片平台	方案名称	模组型号
R328	r328s2_perf2_xr829-tina	XR829
R328	r328s2_perf1-tina	RTL8723DS
R329	r329_evb5_v1-tina	RTL8723DS
R528	r528_evb2-tina	XR829/XR819S
R818	r818_evb2-tina	XR829/RTL8723DS
R818	r818_evb2-tina	XR829/RTL8723DS
D1	d1_nezha-tina	XR829/RTL8723DS/AW869B

注：有些方案在menuconfig配置中，可能模组并没有默认使能打开

4 移植原理概述

本章总体先从蓝牙系统软件架构、移植原理、移植步骤等方面进行阐述说明，使读者对整个蓝牙体系结构以及移植原理有一个初步认识。

4.1 蓝牙系统软件框架

蓝牙是一种短距离无线通信，从整体结构上蓝牙可以分为控制器（Controller）和主机（Host）；控制器主要实现底层射频硬件的链路管理，接口管理等，包括了 Link manager，Baseband controller，Phy 等，；主机（Host）则主要是提供蓝牙应用服务的基础，方便向应用层对蓝牙功能的访问，包括 L2CAP,SMP，SDP，A2DP,HFP,GATT 等。而主机（Host）又是通过 Host Controller interface（简称 HCI）来访问控制器（Controller）的。针对不同的场景，Host 和 Controller 可以运行在一个设备上，也可以运行在不同的设备上。本文章主要阐述的是 Host 和 Controller 运行在不同的设备上。

图 4-1: Host_and_Controller

对于 Host 和 Controller 分别运行在不同的设备上时，就需要通信接口进行连接，常见的通信接口有 USB,SDIO,UART 等。目前大部分模组厂商使用的接口类型都是 uart，所以本文中主要还是以 uart 的接口类型来进行阐述说明。

Tina Linux 主流平台上，Host 软件一般运行在全志平台的主控上，Controller 软件运行在无线模组端。对于应用开发者这来说，不需要关心 Controller 端的软件实现，重点是在 Host 端的软件上。目前 Tina Linux 平台上，Host 端的协议栈主要是采用开源 bluez 协议栈。所以最终的目标就是在主控搭配指定蓝牙模组的平台上，将 bluez 协议栈跑起来。

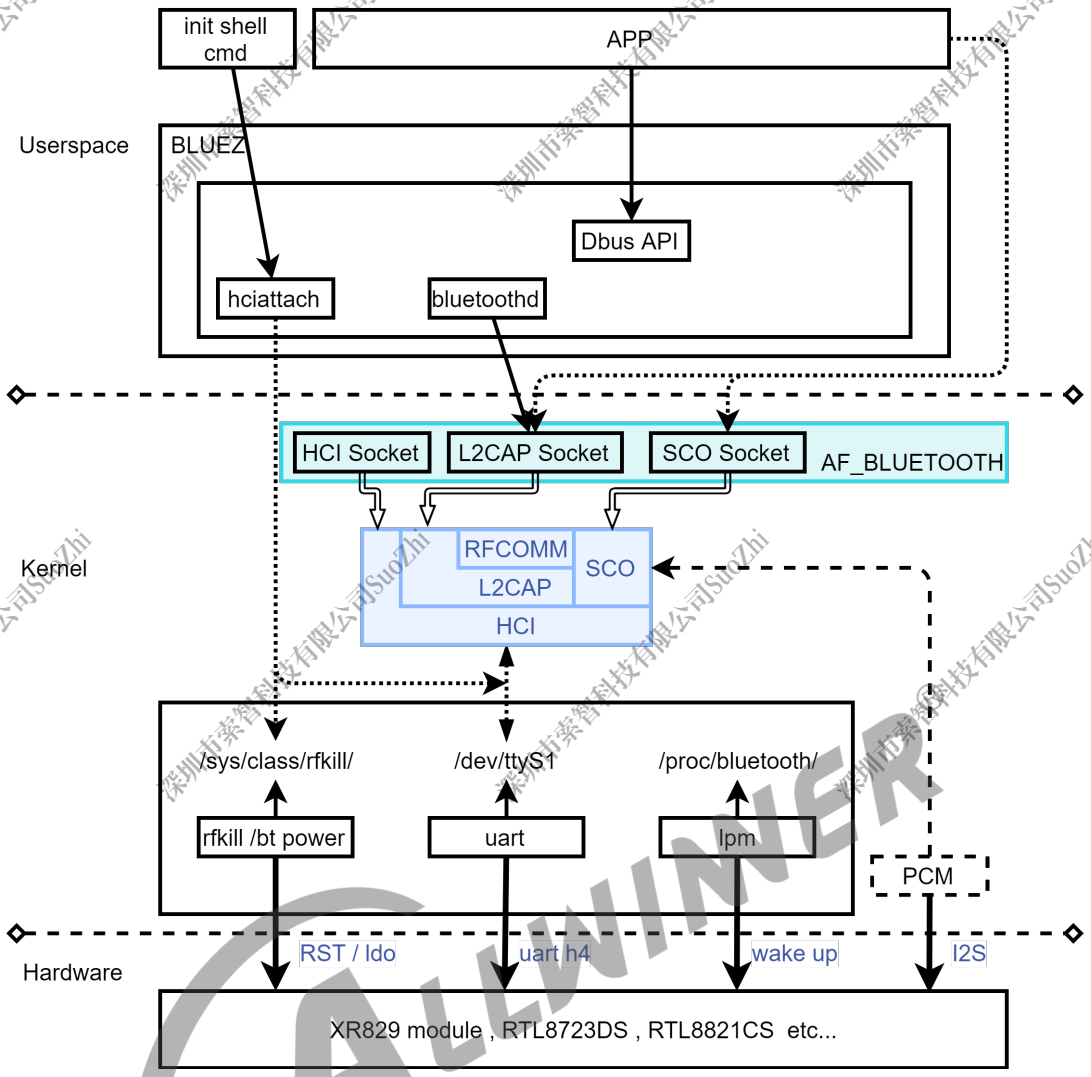

图 4-2: 蓝牙整体软件框架图

软件可以分为几个部分：rfkill, hci uart driver,bt lpm driver,hciattach,firmware，kernel_bluez,bluetoothd

模块	说明
rfkill	Linux 提供用于管理蓝牙卡片供电, 复位使能等功能的子系统
hci uart driver	主控端与蓝牙模组通信的接口驱动
bt lpm driver	用于管理主控休眠唤醒蓝牙模组以及蓝牙模组唤醒主控
hciattach	用于对蓝牙模组进行初始化，包含从主控端下载蓝牙运行需要的固件程序到蓝牙模组端
firmware	Controller 运行需要的固件程序，一般会使用 hciatach 下载到蓝牙模组中
kernel_bluez	包含 HCI,L2CAP,RFCOMM,SCO 等，bluez 协议栈有部分协议在 Linux kernel 中实现
bluetoothd	bluez 协议栈用户空间协议部分

在进行模组移植时，一般模组厂商至少需提供 firmware 和 hciattach。而其他部分模块如果厂商没有定制修改，则可以使用 Linux 标准提供的组件系统。

4.2 移植流程

移植适配蓝牙模组，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。

- 获取模组原厂提供的驱动资料包
- 确认硬件工作的条件配置软件输出
- 使能蓝牙协议栈
- 移植 hciattach
- 调试验证

4.2.1 获取模组厂提供的驱动资料包

在方案上移植一款蓝牙模组时，可以先查看《支持列表》章节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，阅读蓝牙模组正常工作需要满足什么硬件条件，软件上是否支持 bluez 协议栈以及是否有定制修改，如果支持 bluez 协议栈资料包中至少应该包含 hciattach 和 firmware。

4.2.2 确认硬件的工作条件配置软件输出

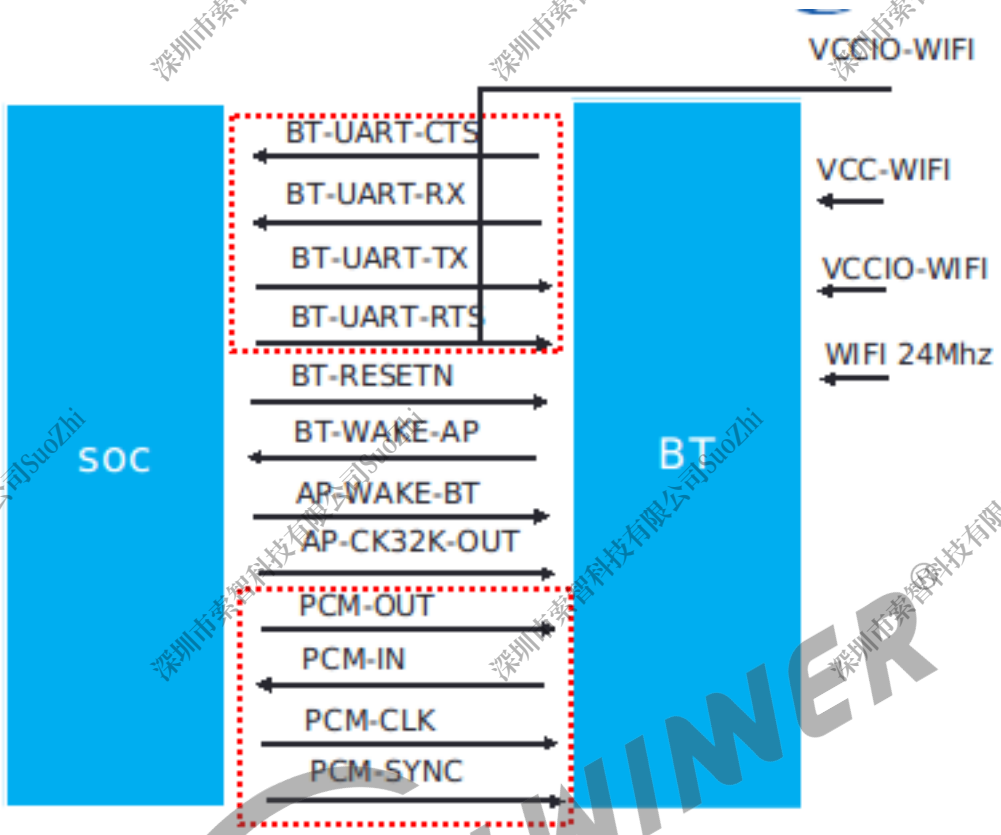

图 4-3: 硬件接线简图

蓝牙模组移植的第一步就是需要满足蓝牙模组启动的硬件工作条件，主流的蓝牙模组和主控的接线图简化如上。可以分为以下几个部分：

功能	说明
供电	一般情况下，蓝牙供电和 WiFi 供电是复用的，分别提供模组的供电以及 IO 通信的上拉。如果供电是从电源管理芯片中给出，那么软件上需要配置电源管理芯片，使能对应的电源域输出稳定电压；如果是直接供电，则软件上一般不需要做配置
时钟	分为主时钟和次时钟，主时钟为蓝牙模块正常工作提供时钟源，可以由外挂晶振或者主控输出 24M/26M 等（根据实际模组规格而定）；次时钟（AP-CK32K-OUT）主要是蓝牙模组的辅助时钟，对于一些模组并不是必须的，常用来在模组休眠提供时钟源。通常情况下主时钟直接由外挂晶振给出，不需要做软件配置；次时钟一般由主控输出，需要软件配置使能
复位使能	主控 GPIO 会输出一路信号用来复位使能蓝牙模块，如图 BT-RESETN；蓝牙要正常工作，需要将其复位拉高

功能	说明
休眠唤醒	分为主控控制蓝牙模块休眠与唤醒（AP-WAKE-BT）和蓝牙模块唤醒主控（BT-WAKE-AP），前者主控通过拉高或拉低使蓝牙模组进入休眠或者唤醒状态；后者主要是在主控进入休眠状态后，蓝牙可以通过中断信号触发主控唤醒。一般需要软件配置将 AP-WAKE-BT 设置为输出模式，BT-WAKE-AP 设置为中断触发输入模式。休眠唤醒在一些模组上如果不需要此功能时，则不需要进行配置，而一些模组在正在工作模式需要把 AP-WAKE-BT 输出拉高，否则蓝牙会进入休眠模式，具体如何操作还需要跟模组原厂进行确认
uart 接口	蓝牙模组和主控数据传输接口，硬件根据不同蓝牙模组要求可能需要外接上拉；通常软件上的只需要选定打开使用的 uart 号即可
PCM 接口	蓝牙语音通话音频传输接口，只有在蓝牙语音通话才会用到此部分

Linux 对无线设备供电使能有一套标准的子系统（rfkill 子系统），关于本小节大部分的软件操作都是围绕 rfkill 子系统来进行，allwinner 对应 rfkill 子系统已经实现，用户一般都不需要再进行修改，大部分工作只需要根据实际的板级来配置设备树即可，关于 rfkill 详细的实现可以参考章节。下面以一款 WiFi 和 BT 二合一的无线模组根据实际原理图来依次阐述硬件的工作条件，软件上如何进行配置。

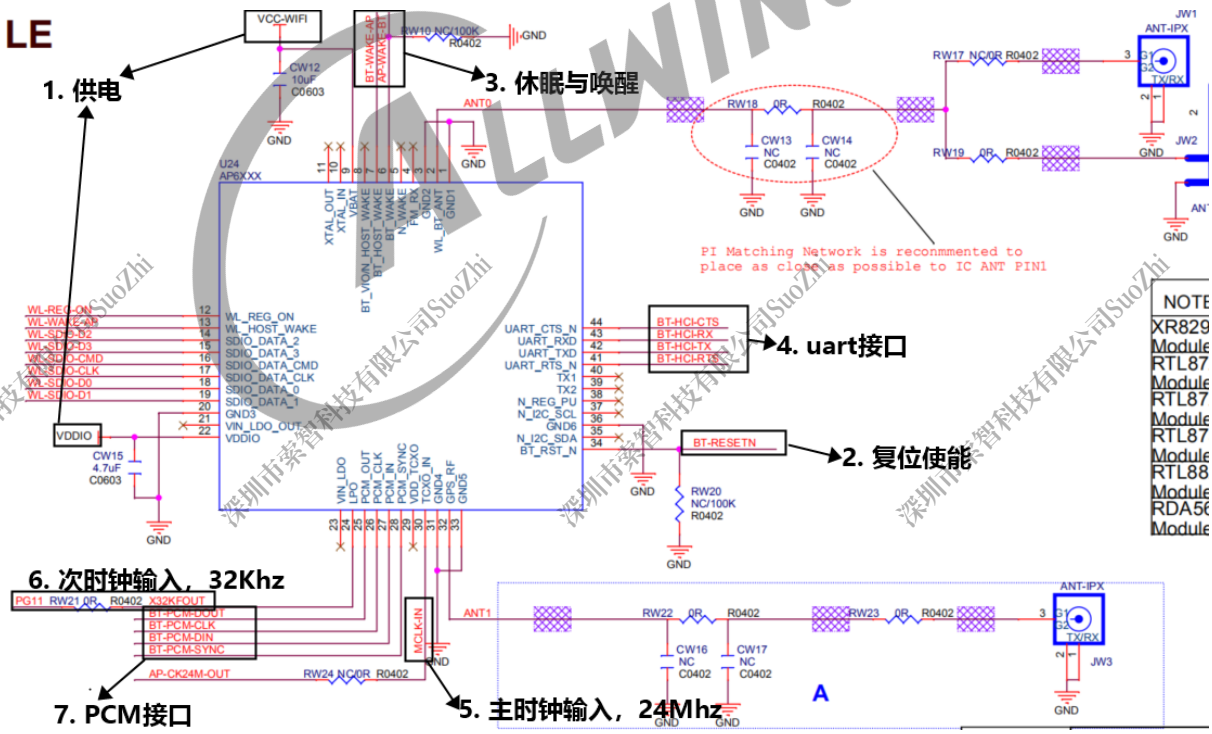

图 4-4: 实际原理图

4.2.2.1 确认供电

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。下面以两路供电来源于 AXP305B 的进行举例说明：

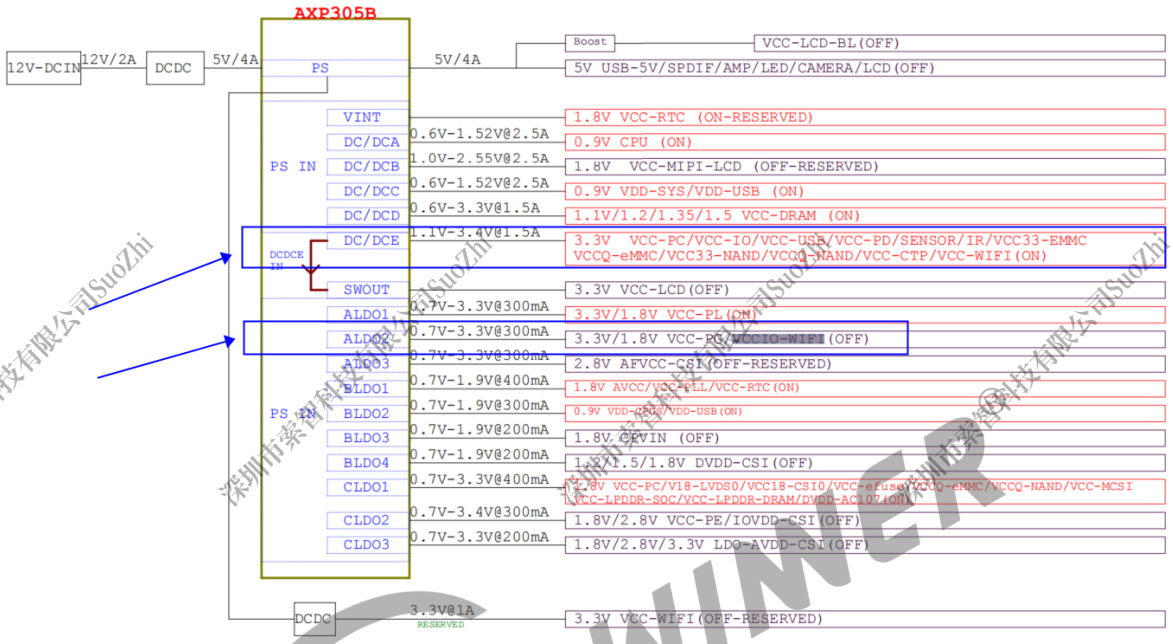

图 4-5: AXP

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，从原理图来判断，两路电源是由电源管理芯片 DC/DCE 和 ALD02 分别输出。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，对于蓝牙开发工程师来说只需要会配置电源输出即可，配置方法见小节关于 bt_power1、bt_io_regulator、bt_io_voltage、bt_power_voltage 标识的配置。

4.2.2.2 确认复位使能 GPIO

蓝牙正常工作时，需要将 BT-RESETN 将 GPIO 配置为输出模式，从低电平拉到高电平，从原理图中确认到 BT-RESETN 连接的是 GPIO PL2，因此设备树中配置 bt_rst_n 标识为 PL2 即可（见小节）。

注意：不同厂家可能对 BT-RESETN 的信号时序有要求，如果默认 sunxi-rf 驱动的实现无法启动蓝牙，需要跟模组厂商进行确认规格需求。

4.2.2.3 休眠与唤醒功能

休眠与唤醒功能大多数情况下与蓝牙模组正常工作是不耦合的，即使这部分不进行移植适配，蓝牙也能正常工作。当然也有部分蓝牙模组存在特殊情况，即使不使用休眠唤醒功能，也需要把 AP-WAKE-BT（见硬件接线简图、实际原理图）进行拉高，原因为如果不拉高，controller 会一直进入休眠状态。下面以 xradio 进行说明配置过程：

（1）获取到 xradio 模组厂商提供的 xradio_btlpm.c。

（2）将该文件放到 linux-xx/drivers/bluetooth 目录下。

（3）修改 Makefie 和 Kconfig 使其能够编译和配置选择。

```
linux-xx/drivers/bluetooth/Makefile
+obj-$(CONFIG_XR_BT_LPM)     += xradio_btlpm.o

linux-xx/drivers/bluetooth/Makefile/Kconfig
+config XR_BT_LPM
+    tristate "Xradio Bluetooth sleep driver support"
+    depends on BT_HCIUART_H4
+    help
+      Bluetooth Bluetooth sleep Driver.
+      This driver provides the dynamic active power saving mechanism for
+      bluetooth radio devices.

+      Say Y here to compile support for bluesleep support into the kernel
+      or say M to compile it as module (bluesleep).
```

（4）添加板级相关配置，xradio_btlpm.c 中将 GPIO 相关板级配置将通过读取设备的方式获取，因此需要在设备树中进行配置板级的 GPIO，主要是对 bt_wake，bt_hostwake，uart_inde 标识填充（见小节）。

（5）通过 3 步骤，就可以进行内核配置选择进行编译。

```
在tina根目录下执行：make kernel_menuconfig
[*] Networking support  --->
   <*>   Bluetooth subsystem support  --->
    Bluetooth device drivers  --->
      <*> Xradio Bluetooth sleep driver support //依赖H4协议，所以需要先选择H4
      <*> HCI UART driver
        [*]   UART (H4) protocol support
```

4.2.2.4 uart 接口配置

蓝牙模组是通过 uart 通信接口与主控进行连接，bluez 协议栈对 uart 的操作都是调用标准 API 接口。在移植蓝牙模组前，需要确保 uart 驱动时正常的（uart 驱动见 uart 驱动开发指南）。通常情况下，uart 已经准备好，可以在设备开机启动之后，通过检查是否存在/dev/ttyS1（uart1）节点来判断 uart 是否已经使能，如果没有使能，可以先查看设备树中关于 uart 是否已经 enable，同时需要确认 uart 的 gpio 是否对应上。

```
uart1: uart@05000400 {
        status = "okay";  //开启uart，如果是disabled则表示未启动。
};
```

如果 dts 配置成功后，uart 依旧未设置成功，则与 uart 工程师进行沟通确认。

4.2.2.5 主时钟输入

主时钟是提供给蓝牙模组工作的时钟源，需要检查原理图时钟源的输入是通过外挂时钟提供还是通过主控芯片输出提供，通常情况下大部分都还是使用外挂时钟提供时钟源，软件不需要进行配置。如果使用的是模块，那么模块厂商多数情况下都已经将晶振直接贴在模块上了。因此可以检查下原理图，确定时钟源来源，检查板子是否已经贴了晶振，必要情况可以通过示波器测量来进行确认。

4.2.2.6 次时钟输入

次时钟，主要是提供模组 32.768kHz 的时钟源，通过检查原理图确认时钟源的输入是主控提供还是外接晶振提供，如果直接外接了晶振，那么软件也不需要在进行配置。如果需要主控输出，那么就需要配置 CLK 驱动，使其输出时钟。关于主控端 CLK 驱动需由 CLK 驱动开发工程师先完成，完成之后，只需要在设备树中引用即可。在设备树中，主要是通过 clocks 符号来进行引用选择那一路时钟源，sunxi-rf 驱动中将会解析该符号标识，调用 devm_clk_get 函数进行使能输出（见章节）。

4.2.2.7 PCM 接口配置

PCM 接口主要是用于蓝牙语音通话功能，该部分后续再进行补充。

4.2.3 sunxi-rf 驱动

Linux 提供了标准的 rfkill 子系统来管理射频的供电使能，主控厂商需要根据标准的 rfkill 子系统来进行注册实现 WiFi 或 BT 的操作。在 Tina Linux 平台中，关于蓝牙的 rfkill 实现在 linux-x.x/drivers/misc/sunxi-rf/sunxi_bluetooth.c 中。下面是配置内核的使能方法：

```
make kernel_menuconfig
 /*allwinner rfkill实例实现*/
 Device Drivers  --->
   Misc devices  --->
    <*> Allwinner rfkill driver

 /*Networking rfkill0 switch 子系统使能*/
 [*] Networking support  --->
   <*>   RF switch subsystem support  --->
     [ ] RF switch input support //这个不能选
```

```
    <*> GPIO RFKILL driver
```

Linux 驱动遵循设备驱动模型，所以自 Linux3.4 内核以上，板级相关的配置归纳到设备树 dts 中实现。驱动程序会读取解析设备树的配置标识，实现多平台方案核心驱动不用改动，差异点只需要各平台的设备树即可。

4.2.3.1 蓝牙设备树配置

dts 文件路径，如 R328 在/tina/device/config/chips/r328s2/configs/std/board.dts，除此之外还有该路径下的 sys_config.fex 文件，其优先级更高。除了这两个文件可以配置外，还有个原始方案的基本配置/tina/lichee/linux-4.9/arch/arm/boot/dts/方案代号.dtsi。

```
bt: bt@0 {
    compatible    = "allwinner,sunxi-bt";          蓝牙供电和使能的属性名称，一般不需要修改
    clocks        = <&clk_losc_out>;               主控输出给32khz信号引用，对应硬件接线简图中的AP
    -CK32K-OUT
    bt_power_num = <0x01>;                         使用电源管理芯片供电蓝牙模块的供电数目，如果不使
    用电源管理芯片可忽略
    bt_power1      ="axp806-dcdce";                使用电源管理芯片DCDCE域输出的主电源，对应硬件接
    线简图中的VCC-WIFI
    bt_io_regulator = "axp806-aldo2";              使用电源管理芯片ALDO2域输出的上拉电源，对应硬件
    接线简图中的VCCIO-WIFI
    bt_io_voltage = <3300000>;                     使用电源管理芯片输出的上拉电源默认电压值
    bt_power_voltage =<330000>;                    使用电源管理芯片输出的主电源默认电压值
    bt_rst_n      = <&r_pio PL 2 1 0x1 0x2 0>;     蓝牙复位使能GPIO，对应硬件接线简图中的BT-
    RESETN
    status        = "okay";
};

btlpm: btlpm@0 {
    compatible  = "allwinner,sunxi-btlpm";
    uart_index  = <0x1>;                           使用的uart序号
    bt_wake     = <&r_pio PL 4 1 0x1 0x2 1>;       主控控制蓝牙模块休眠与唤醒GPIO，对应硬件简图AP-
    WAKE-BT，设置为输出模式
    bt_hostwake = <&r_pio PL 3 6 0x1 0x2 1>;       蓝牙模块唤醒主控GPIO，对应硬件简图BT-WAKE-AP，
    设置为中断输入模式
    status      = "okay";
};
```

4.2.3.2 驱动简析

下面我们来分析下 sunx-rf 驱动中，是如何读取设备树中的配置进而使能供电和 GPIO 的，代码位于 linux-x.x/drivers/misc/sunxi-rf/sunxi_bluetooth.c（如果不感兴趣可以跳过）：

（1）解析 dts 配置 Linux 设备和驱动实现是分离的，平台相关的配置在设备树中实现，这里简单描述下 sunxi-rf 驱动如何解析设备树 board.dts。

```
int sunxi_bt_probe(struct platform_device *pdev)
{
  /*1. 获取主电源供电的电源数目，对应的是VCC-WIFI，只有一路则dts中应该配置bt_power_num = 1*/
  of_property_read_u32(np, "bt_power_num", &val)
```

```
  /*2. 从bt_power_voltage和bt_io_voltage标识读取电压值*/
  of_property_read_u32(np, "bt_power_voltage", &val)
  of_property_read_u32(np, "bt_io_voltage", &val)

  /*3. 根据1步骤获取的主电源数进行遍历各个电源域的名称，对应bt_power1*/
  for(i = 0; i < (data->power_num); i++) {
    of_property_read_string(np, bt_name_buf, &power)
    strcpy(data->bt_power_name[i], power);
  }

  /*4. 获取IO上拉的电源域名称，对应VCCIO-WIFI，dts标识为bt_io_regulator*/
  of_property_read_string(np, "bt_io_regulator", &io_regulator)


  /*5. 解析BT-RESETN引脚标识，配置GPIO模式，默认先拉低*/
  data->gpio_bt_rst = of_get_named_gpio_flags(np, "bt_rst_n", 0,(enum of_gpio_flags *)&
    config);
  ret = devm_gpio_request(dev, data->gpio_bt_rst, "bt_rst");
  ret = gpio_direction_output(data->gpio_bt_rst, 0);

  /*6. 解析次时钟源时钟域，并使能*/
  of_property_read_string(np, "clocks", &clocks)
  data->clk_name = devm_kzalloc(dev, 64, GFP_KERNEL);
  strcpy(data->clk_name, clocks);
  data->lpo = devm_clk_get(dev, data->clk_name);
  ret = clk_prepare_enable(data->lpo);

  /*7. 注册rfkill驱动*/
  /*
  * linux系统提供rfkill子系统用来管理无线模块供电使能的通用接口，驱动注册成功后，
  * 用户就可以通过/dev/rfkill和属性文件来操作蓝牙进行上下电，如下:
  * 上电: echo 0 >/sys/class/rfkill/rfkill0/state
  * 下电: echo 1 >/sys/class/rfkill/rfkill0/state
  * 关于上下电的实际操作就在sunxi_bt_rfkill_ops函数集合中实现，调用到sunxi_bt_on函数
  */
  data->rfkill = rfkill_alloc("sunxi-bt", dev, RFKILL_TYPE_BLUETOOTH,&sunxi_bt_rfkill_ops,
    data);
  rfkill_set_states(data->rfkill, true, false);
  ret = rfkill_register(data->rfkill);
}
```

（2）上下电操作前面说到 Linux 系统提供了 rfkill 子系统来规范无线模块的上下电等使能操作，用户可以通过操作文件节点来进行控制。

命令	功能
echo 0 >/sys/class/rfkill/rfkill0/state	下电
echo 1 >/sys/class/rfkill/rfkill0/state	上电

当用户操作 rfkill 节点时，对应操作的时 sunxi-rf 驱动中 sunxi_bt_on 函数，接下来就简单进行分析该函数的实现原理。

```
static int sunxi_bt_on(struct sunxi_bt_platdata *data, bool on_off)
{
  /*1. 如果是上电操作，则初始值先将BT-RESETN GPIO先拉低*/
  if (!on_off && gpio_is_valid(data->gpio_bt_rst))
```

```
    gpio_set_value(data->gpio_bt_rst, 0);
  /*2. 根据sunxi_bt_probe函数中解析的主电源数进行遍历处理*/
  for (i = 0; i < (data->power_num); i++) {
    /*2.1 获取电源域句柄*/
    data->bt_power[i] = regulator_get(dev, data->bt_power_name[i]);
    /*2.2 如果是上电操作，则enable对应电源域，并设置电压值*/
    if(on_off) {
      regulator_enable(data->bt_power[i]);
      regulator_put(data->bt_power[i]);
      ret = regulator_get_voltage(data->bt_power[i]);
      if (ret < data->bt_power_voltage) {
        regulator_set_voltage(data->bt_power[i],data->bt_power_voltage,data->
    bt_power_voltage);
      }
    /*2.3 如果是下电操作，则disable对应电源域，进行下电*/
    }else {
      regulator_disable(data->bt_power[i]);
    }
    regulator_put(data->bt_power[i]);
  }

  /*3. 设置上拉的电源的值，逻辑也是根据是上电操作还是下电操作来处理，同2*/
  ...

  /*4. BT-RESETN GPIO拉高*/
  if (on_off && gpio_is_valid(data->gpio_bt_rst)) {
    mdelay(10); //一般拉高需要一定时序，所以延时10ms
    gpio_set_value(data->gpio_bt_rst, 1);
  }
}
```

4.2.4 使能蓝牙协议栈

从蓝牙整体软件框架图中，bluez 协议栈一部分在用户空间实现，一部分在内核中实现。协议栈的部分在 Tina Linux 移植中已经移植了，对于用户来说，只需要在编译之前配置选择上即可。下面是配置选择方法：

协议栈内核空间部分选择

```
make kernel_menuconfig
[*] Networking support  --->
  <*>   Bluetooth subsystem support  --->
    [*]   Bluetooth Classic (BR/EDR) features //经典蓝牙特性支持
    < >     RFCOMM protocol support   //如果需要SPP文件传输等相关功能可以选上
    [*]   Bluetooth Low Energy (LE) features  //低功耗蓝牙特性支持
    [*]   Export Bluetooth internals in debugfs //蓝牙调试节点
      Bluetooth device drivers  --->
        <*> HCI UART driver
          -*-   UART (H4) protocol support  //hci uart h4驱动
```

协议栈用户空间部分选择

```
make menuconfig
Utilities  --->
  -*- bluez-daemon..........................................  Bluetooth daemon  //协议栈核心
    进程
```

```
    [*] Enable xr829 extra config //如果模组是XR829则选上
  -*- bluez-utils..... Bluetooth utilities //协议栈调试
      工具
  -*- bluez-utils-extra..... Bluetooth additional utilities
```

下面列出协议栈的代码路径，开发能力强的开发者可以进行阅读，必要情况下可以进行定制修改。用户空间的协议栈部分，构建编译规则遵循 openwrt 构建法则，感兴趣的可以看下 openwrt 的官网先熟悉下 packge 的编译方式，主要是分为源码包、构建编译规则包，编译输出包三部分，如下：

软件包	说明
tina/dl/bluez-5.54.tar.xz	软件源码包
tina/package/utils/bluez	控制编译逻辑: 解压 bluez-5.54.tar.xz，进行编译生成可执行文件，再将对应文件和配置拷贝到文件系统中
tina/out/具体方案/compile_dir/target/bluez-5.54	实际编译的目录

内核的协议栈部分，如下：

路径	说明
lichee/linux-xx/net/bluetooth	协议核心代码，主要实现 HCI,L2CAP 以及跟用户空间交换的 socket 等，这部分也是 Linux 内核社区代码，一般情况下都不需要更改
drivers/bluetooth	厂商驱动适配层代码，包含各模组厂商的休眠唤醒驱动、hci uart 适配驱动等等

4.2.5 移植 hciattach

蓝牙模组初始化主要是通过 hciattach 进行完成，hciattch 依次完成 uart 接口初始化、蓝牙固件下载、传输波特率协商、复位启动等功能。一般该部分有模组厂商进行提供，不同的厂商实现方式有所差别，一般两种方式：基于原生 bluez 协议栈的 tools 工具进行适配生成 hciattach 和单独给出 hciattach 的软件包，前者需要将补丁打到 bluez 协议栈中编译生成 hciattach 可执行文件，后者只需要使用平台的交叉编译工具链直接编译生成可执行文件即可。

```
tina/out/r328s2-xxx方案/compile_dir/target/bluez-5.54/tools/hciattach.c
```

目前 Tina Linux 平台已经适配支持 Xadio，realteak 厂商部分模组的 hciattach，用户只需要进行配置选择即可，此外其他厂商的如果没有先进行适配，则需要跟模组厂商获取 hciattach 的软件包，依据情况进行适配生成 hciatach 可以执行文件。

```
make menuconfig
Utilities  --->
  rtk_hciattach  --->
    <*> rtk_hciattach.................. Realtek BT HCI UART initialization tools
    //realteak模组厂商提供的hciattch，该厂商是以独立的软件包方式提供，选择该配置文件后，编译就会生成
    rtk_hciattach的可执行文件
  -*- bluez-daemon.......................................... Bluetooth daemon
  [*] Enable xr829 extra config
    //xradio模组厂商提供的hciattch，是在原生bluez协议栈上提交的补丁，选择该配置时候，会编译生成hciattch
    可执行文件
```

4.2.6 调试验证

硬件的工作条件、协议栈使能、以及 hciattach 都准备好之后，就可以进行编译调试了，整体编译打包生成固件烧录到设备中。就可以按照下面的启动部分进行测试验证：

```
1. 硬件工作条件：供电使能
echo 0 > /sys/class/rfkill/rfkill0/state //蓝牙复位
echo 1 > /sys/class/rfkill/rfkill0/state //蓝牙上电使能
2. 蓝牙初始化
hciattach -n ttyS1 xradio & //以xr829为例，不同的模组厂商，命令不一样。
3. 启动协议栈
bluetoothd -n -d &
4. 网卡启动
hciconfig hci0 up
5. 验证功能
查询状态：hciconfig -a
扫描：hcitool -i hci0 lescan
```

在通过软件配置完成硬件的工作条件之后，在设备运行起来之后，如果无法正常运行蓝牙设备，我们则需要通过辅助设备如万用表、示波器来逐一进行排查确认，所有的工作条件是否满足，如果不满足，则需要仔细检查 board.dts 的配置文件是否正确，必要情况下，需要跟踪 sunxi-rf 驱动模块代码的实现，确认是否符号预期。

以下是软件启动流程，便于开发者进行对照分析：

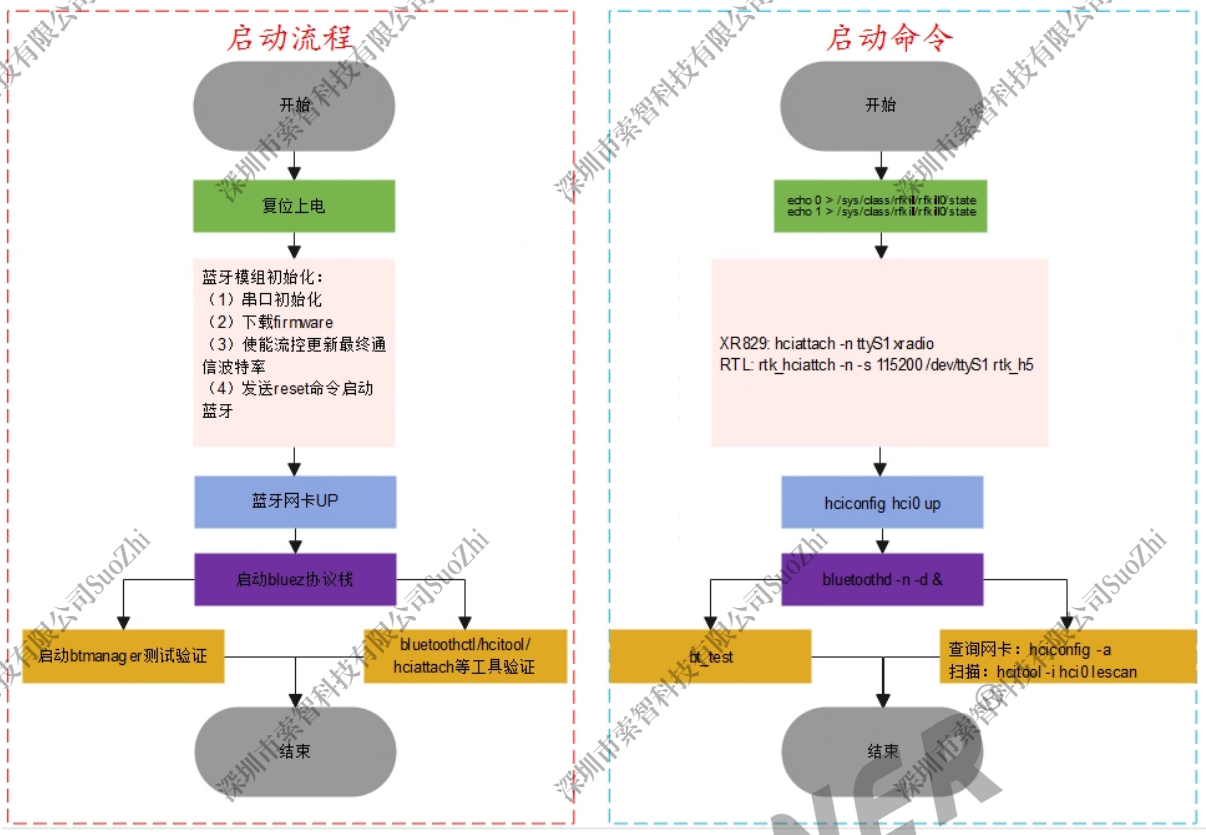

图 4-6: 启动流程

5 新模组移植示例

关于模组移植的原理，请参考《移植原理概述》章节。本章节将分别选择 D1+AW869B,R528+XR819S 两个平台进行说明 AW869B,XR819S 的移植过程。

5.1 AW869B(AIC8800) 模组移植

```
主控：D1
无线模组：AW869B(AIC8800)
内核版本：linux-5.4
方案：d1-nezha
```

5.1.1 获取模组原厂提供的驱动资料包

模组原厂一般会提供基础的资料包，这里通过模组原厂的技术接口获取到资料包，如下：

```
.
├── AIC8800_SDIO_porting_guide_v1_1.pdf
└── driver_fw
    ├── aic
    │   ├── libbt
    │   │   ├── Android.mk
    │   │   ├── firmware
    │   │   ├── gen-buildcfg.sh
    │   │   ├── include
    │   │   ├── OWNERS
    │   │   ├── src
    │   │   └── vnd_buildcfg.mk
    │   ├── rftest-tools
    │   │   ├── aicrf_test_apk_v3.0.pdf
    │   │   ├── Android.mk
    │   │   └── app-debug.apk
    │   └── wlan
    ├── driver
    │   └── aic8800
    │       ├── aic8800_bsp
    │       ├── aic8800_btlpm
    │       ├── aic8800_fdrv
    │       ├── Kconfig
    │       └── Makefile
    └── fw
        ├── aic_userconfig.txt
        ├── fmacfw.bin
        ├── fmacfw_rf.bin
        ├── fw_adid.bin
```

```
├── fw_adid_u03.bin
├── fw_patch.bin
├── fw_patch_table.bin
├── fw_patch_table_u03.bin
├── fw_patch_test.bin
└── fw_patch_u03.bin
```

从上面的资料包来进行分析，资料包主要适用于 Android 部分，并不适用于 Linux 平台，但对于 Tina Linux 平台蓝牙用的是标准 bluez 协议栈，最终目的需要把蓝牙卡片初始化起来，通常情况下需要把 firmware 下载到卡片中，发送 hci 复位、同步等指令。初步判断 AW869 蓝牙的固件下载跟 WiFi 的固件下载是一起的，在 aic8800_bsp 中实现，因此蓝牙要启动，需要先将 WiFi 移植完成，先将固件下载到模组中。固件下载完成之后，使用 hciattach 进行与模组同步，就可以将 hci0 带起来了。本章节不会说明 WiFi 部分的移植，关于 WiFi 的移植请参考 WiFi 移植文档。

5.1.2 确认硬件的工作条件配置软件输出

根据平台的硬件原理图，进行分析主控 (D1) 与模组 (AW869) 硬件连线情况，主要梳理供电电源、上拉电源、复位使能 GPIO、休眠与唤醒 GPIO、UART GPIO、主次时钟。AW869B 的原理图引脚图如下所示。

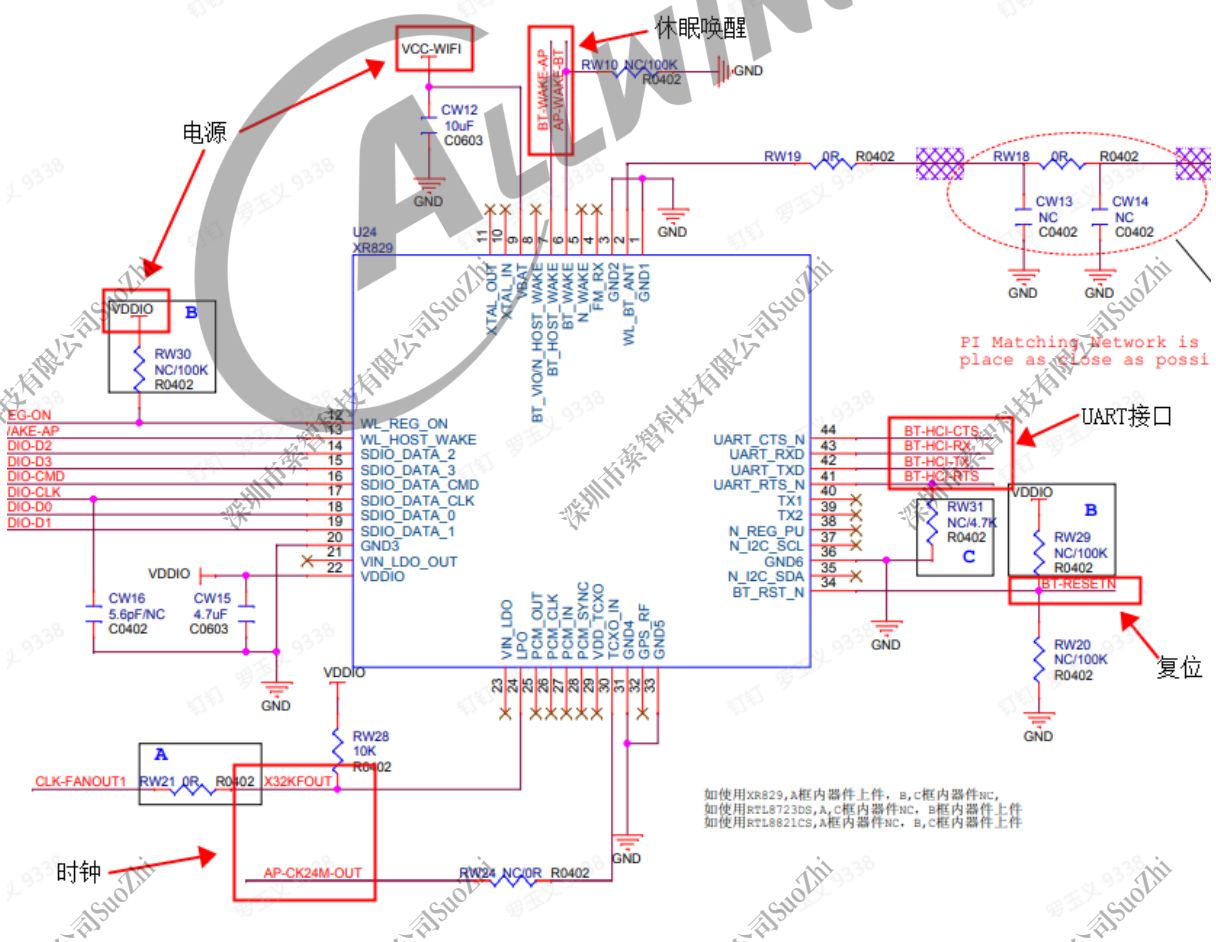

图 5-1: AW869B 原理图

下面将依次进行确认。

5.1.2.1 电源

根据原理图找到 AW869B 的 VCC-WIFI 和 VDDIO 的连接方式。VCC-WIFI 属于 D1 电源树的 DC/DC4，其默认开启 3.3V，不需要额外配置；VDDIO 对应 VCC-PG，根据芯片手册其电压需要配置到 1.8V。

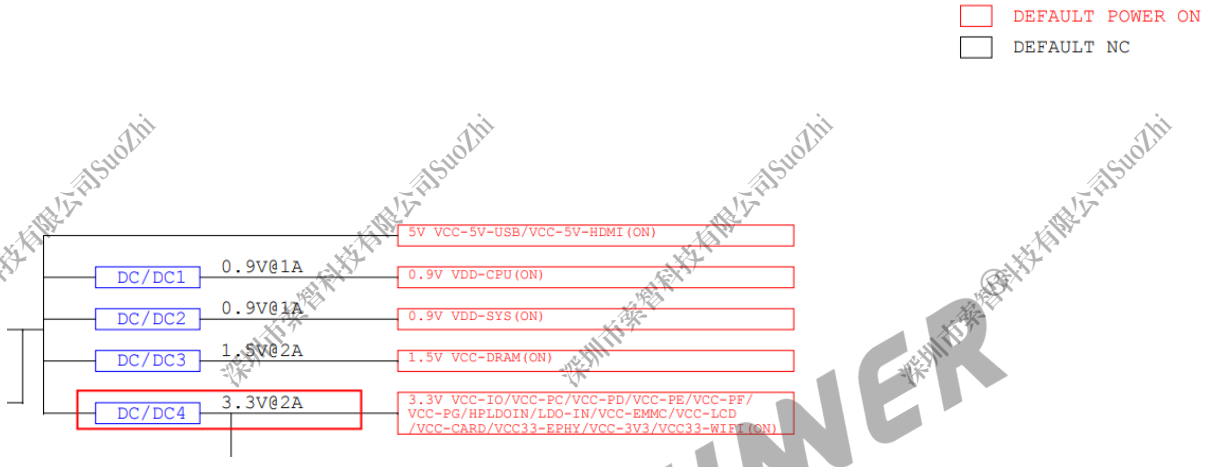

图 5-2: D1 电源树

VCC-PG 的配置在 tina/device/config/chips/d1/configs/nezha/board.dts

```
vcc-pg-supply = <&reg_pio1_8>;
```

5.1.2.2 复位使能

根据原理图中找到 BT-RESETN 连接的是 GPIO PG18，因此设备树中配置 bt_rst_n 标识为 GP18。

配置文件路径 tina/device/config/chips/d1/configs/nezha/board.dts

```
bt: bt@0 {
    compatible    = "allwinner,sunxi-bt";
    clock-names = "32k-fanout1";
    clocks = <&ccu CLK_FANOUT1_OUT>;
    /*bt_power_num = <0x01>;*/
    /*bt_power      = "axp803-dldo1";*/
    /*bt_io_regulator = "axp803-dldo1";*/
    /*bt_io_vol = <3300000>;*/
    /*bt_power_vol = <330000>;*/
    bt_rst_n      = <&pio PG 18 GPIO_ACTIVE_LOW>;   //复位引脚配置
    status        = "okay";
};
```

5.1.2.3 休眠唤醒

休眠唤醒暂时不支持，这里不做阐述。

5.1.2.4 UART 接口配置

根据原理图中找到 UART 引脚 PG06 PG07 PG08 PG09，在设备树里进行配置并使能。

配置文件路径是 tina/device/config/chips/d1/configs/nezha/board.dts

```
uart1_pins_a: uart1_pins@0 {  /* For EVB1 board */
pins = "PG6", "PG7", "PG8", "PG9";
function = "uart1";
drive-strength = <10>;
bias-pull-up;
};

uart1_pins_b: uart1_pins {  /* For EVB1 board */
pins = "PG6", "PG7", "PG8", "PG9";
function = "gpio_in";
};

&uart1 {
    pinctrl-names = "default", "sleep";
    pinctrl-0 = <&uart1_pins_a>;
    pinctrl-1 = <&uart1_pins_b>;
    status = "okay";
};
```

5.1.2.5 主次时钟

根据原理图看出主时钟由外部 24M 晶振提供，次时钟是主控输出 32k 时钟。

配置文件路径是 tina/device/config/chips/d1/configs/nezha/board.dts

```
bt: bt@0 {
    compatible  = "allwinner,sunxi-bt";
    clock-names = "32k-fanout1";        //主控输出32k时钟名字
    clocks = <&ccu CLK_FANOUT1_OUT>;    //主控输出32k时钟
    /*bt_power_num = <0x01>;*/
    /*bt_power      = "axp803-dldo1";*/
    /*bt_io_regulator = "axp803-dldo1";*/
    /*bt_io_vol = <3300000>;*/
    /*bt_power_vol = <330000>;*/
    bt_rst_n      = <&pio PG 18 GPIO_ACTIVE_LOW>;
    status        = "okay";
};
```

5.1.3 sunxi-rf 驱动

使能 sunx-rf 驱动，用于解析上一章节中 dts 中的配置项

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

5.1.4 使能蓝牙协议栈

协议栈分为内核层和用户层，这部分只需要配置选择即可，协议栈内核空间配置选择如下：

```
make kernel_menuconfig
[*] Networking support  --->
  <*>   Bluetooth subsystem support  --->
    [*]   Bluetooth Classic (BR/EDR) features //经典蓝牙特性支持
    < >     RFCOMM protocol support   //如果需要SPP文件传输等相关功能可以选上
    [*]   Bluetooth Low Energy (LE) features  //低功耗蓝牙特性支持
    [*]   Export Bluetooth internals in debugfs //蓝牙调试节点
      Bluetooth device drivers  --->
        <*> HCI UART driver
          -*-   UART (H4) protocol support  //hci uart h4驱动
```

协议栈用户空间配置选择如下：

```
make menuconfig
Utilities  --->
  -*- bluez-daemon..................................... Bluetooth daemon  //协议栈核心
    进程
    [*] Enable xr829 extra config //如果模组是XR829则选上
  -*- bluez-utils...................................... Bluetooth utilities  //协议栈调试
    工具
  -*- bluez-utils-extra.............................. Bluetooth additional utilities
```

5.1.5 启动初始化

启动初始化大部分厂商都是通过 hciattach 来实现，启动流程可以参考。对于 AW869B 模组实际上也大同小异，区别在于其固件下载是同 WiFi 一起通过 SDIO 进行的，所以免去的固件下载的步骤，需要在 WiFi 驱动中使能蓝牙支持即可（这里看起来模组厂提供的驱动并不规范，蓝牙和 WiFi 的耦合在一起了）。

```
make kernel_menuconfig
> Device Drivers
 > Network device support
  > Wireless LAN
    <*> AIC wireless Support
     ├── <M> AIC8800 bluetooth Support
```

5.1.5.1 firmware

设备在启动的时候需要从文件系统中读取 firmware 文件，所以需要在编译的时候将 firmware 文件配置到文件系统中，根据提供的资料包里的固件，在 tina/package/firmware/linux-firmware/添加 aic8800 需要的 firmware。

```
tina/package/firmware/linux-firmware/aic8800
├── aic_userconfig.txt    //射频功率校准文件
├── 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，该文件在系统编译的时候，就会将 firmware 拷贝到文件系统中

```
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)/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))
```

注意：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。

```
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
#define AIC_FW_PATH      "/lib/firmware"
```

配置完成之后，m menuconfig 就可以看到新添加的 firmware 配置。注意要取消别的方案的

firmware。

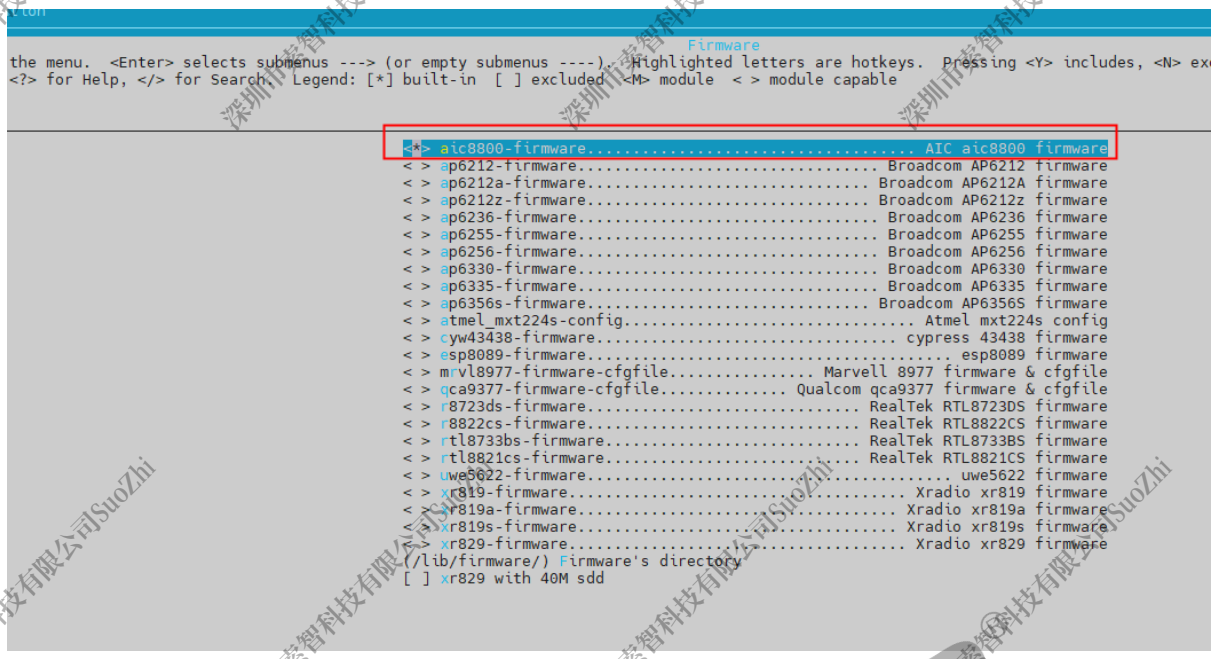

图 5-3: 配置 Firmware

5.1.5.2 移植 hciattach

模组厂商或者驱动人员会提供 hciattach.c 和模组单独的 hciattach_aic.c，这两个文件用于 hciattach 初始化。

注意 AW869b 的 firmware 是通过 bsp 驱动在模块上电的时候下载进去的，所以在 AW869b 的 hciattach_aic 里面是没有下载固件过程的。整个移植过程大致分为四个步骤：

第一步：在目录 tina/out/d1-nezha/compile_dir/target/bluez-5.54/tools/hciattach.c 中添加一组参数到 uart_t 结构体类型的数组里面。

```
1120
1121    /* Xradio XR829 */
1122    { "xradio",     0x0000, 0x0000, HCI_UART_H4,    115200, 1500000,
1123            0, DISABLE_PM, NULL, xradio_xr829, NULL },
1124
1125    { "xr829",      0x0000, 0x0000, HCI_UART_H4,    115200, 1500000,
1126            0, DISABLE_PM, NULL, xradio_xr829, NULL },
1127
1128    { "xr819s",     0x0000, 0x0000, HCI_UART_H4,    115200, 1500000,
1129            0, DISABLE_PM, NULL, xradio_xr819s, NULL },
1130
1131    { "ath3k",      0x0000, 0x0000, HCI_UART_ATH3K, 115200, 115200,
1132            FLOW_CTL, DISABLE_PM, NULL, ath3k_ps, ath3k_pm },
1133
1134    /* QUALCOMM BTS */
1135    { "qualcomm",   0x0000, 0x0000, HCI_UART_H4,    115200, 115200,
1136            FLOW_CTL, DISABLE_PM, NULL, qualcomm, NULL },
1137
1138    /* Intel Bluetooth Module */
1139    { "intel",      0x0000, 0x0000, HCI_UART_H4,    115200, 115200,
1140            FLOW_CTL, DISABLE_PM, NULL, intel, NULL },
1141
1142    /* Three-wire UART */
1143    { "3wire",      0x0000, 0x0000, HCI_UART_3WIRE, 115200, 115200,
1144            0, DISABLE_PM, NULL, NULL, NULL },
1145
1146    /* AMP controller UART */
1147    { "amp",    0x0000, 0x0000, HCI_UART_H4, 115200, 115200,
1148            AMP_DEV, DISABLE_PM, NULL, NULL, NULL },
1149
1150    /* AW869B */
1151    { "aic",        0x0000, 0x0000, HCI_UART_H4, 1500000, 1500000,
1152            FLOW_CTL, DISABLE_PM, NULL, aic_init, aic_post},
1153
1154    { NULL, 0 }
1155 };
```

图 5-4: AW869B-hciattach

第二步：将 hciattach_aic.c 复制到 tina/out/d1-nezha/compile_dir/target/bluez-5.54/tools/目录下，这个文件是 aw869b 具体的初始

化函数，在 tina/out/d1-nezha/compile_dir/target/bluez-5.54/tools/makefile 中添加 hciattach_aic.c 相关的编译中间件。

第三步：在 tina/package/utils/bluez 目录下执行编译命令：mm，注意不要加-B，然后回到根目录下编译打包成固件烧录验证。根据测试的异常 log 分析原因。

第四步：编译生成 hciattach 后，按照后小节进行测试，测试通过之后，在 tina/out/d1-nezha/compile_dir/target/bluez-5.54/路径下生成 patch 放在 tina/package/utils/bluez/patches 文件夹下面。

5.1.6 调试验证

配置完编译打包生成固件烧录到设备中，准备测试验证 AW869 蓝牙功能。

5.1.6.1 分步骤测试验证：

```
1. 硬件工作条件：供电使能
echo 0 > /sys/class/rfkill/rfkill0/state
echo 1 > /sys/class/rfkill/rfkill0/state //蓝牙上电使能
2. 蓝牙初始化
hciattach -s 1500000 /dev/ttyS1 aic &
hciconfig -a //查询状态 DOWN状态
3. 网卡启动
hciconfig hci0 up
hciconfig -a //查询状态 UP状态
4. 启动协议栈
bluetoothd -n -d &
5. 验证功能
hcitool -i hci0 scan //扫描附近蓝牙设备
```

按照上面的指令在终端执行正常 log 如下：

```
root@TinaLinux:/# echo 0 > /sys/class/rfkill/rfkill0/state
[   83.099248] sunxi-rfkill soc@3000000:rfkill@0: block state already is 1
root@TinaLinux:/# echo 1 > /sys/class/rfkill/rfkill0/state      //蓝牙上电使能
[   88.579526] sunxi-rfkill soc@3000000:rfkill@0: set block: 0
[   88.595900] sunxi-rfkill soc@3000000:rfkill@0: bt power on success
root@TinaLinux:/# hciattach -s 1500000 /dev/ttyS1 aic &         //蓝牙初始化
root@TinaLinux:/# AIC Bluetooth init uart with init speed:1500000, final_speed:1500000,
    type:HCI UART H4
AIC Bluetooth: aic_send_hci_cmd
AIC Bluetooth: 03 0c 00
AIC Bluetooth: Received event, len: 7
AIC Bluetooth: 04 0e 04 05 03 0c 00
AIC Bluetooth: Setting local bd addr to 10:11:12:13:14:15
AIC Bluetooth: aic_send_hci_cmd
AIC Bluetooth: 70 fc 06 15 14 13 12 11 10
AIC Bluetooth: Received event, len: 7
AIC Bluetooth: 04 0e 04 05 70 fc 00
AIC Bluetooth: aic_send_hci_cmd
AIC Bluetooth: 4d fc 68 00 d7 18 00 00 f7 18 00 40 00 00 00 28
AIC Bluetooth: 00 90 01 90 01 03 00 02 00 03 00 02 00 28 00 00
AIC Bluetooth: 02 14 00 15 00 14 00 20 00 00 00 08 00 00 00 ff
AIC Bluetooth: ff ff ff 00 00 00 00 20 4e 00 00 13 01 00 00 02
AIC Bluetooth: 73 06 20 07 00 40 00 48 00 47 00 20 00 00 02 a4
AIC Bluetooth: 01 64 00 64 00 08 00 18 00 28 00 8c 00 00 00 40
AIC Bluetooth: 00 00 00 20 4e 00 00 32 00 00 00
AIC Bluetooth: Received event, len: 7
AIC Bluetooth: 04 0e 04 05 4d fc 00
AIC Bluetooth: aic_send_hci_cmd
AIC Bluetooth: 50 fc 01 02
AIC Bluetooth: Received event, len: 7
AIC Bluetooth: 04 0e 04 05 50 fc 00
AIC Bluetooth: aic_send_hci_cmd
AIC Bluetooth: 51 fc 01 01
AIC Bluetooth: Received event, len: 7
AIC Bluetooth: 04 0e 04 05 51 fc 00
AIC Bluetooth: aic_send_hci_cmd
AIC Bluetooth: 52 fc 01 01
AIC Bluetooth: Received event, len: 7
AIC Bluetooth: 04 0e 04 05 52 fc 00
AIC Bluetooth post process
AIC Bluetooth: Done setting line discpline
```

```
Device setup complete

[1]+  Done                    hciattach -s 1500000 /dev/ttyS1 aic
root@TinaLinux:/# hciconfig -a              //查询状态 DOWN状态
hci0:   Type: Primary  Bus: UART
        BD Address: 10:11:12:13:14:15  ACL MTU: 1021:9  SCO MTU: 255:4
        DOWN
        RX bytes:676 acl:0 sco:0 events:37 errors:0
        TX bytes:435 acl:0 sco:0 commands:37 errors:0
        Features: 0xbf 0xee 0xcd 0xfe 0xd8 0x3f 0x7b 0x87
        Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV3
        Link policy: RSWITCH SNIFF
        Link mode: SLAVE ACCEPT

root@TinaLinux:/# hciconfig hci0 up          //启动hci0
root@TinaLinux:/# hciconfig -a              //查询状态 up状态
hci0:   Type: Primary  Bus: UART
        BD Address: 10:11:12:13:14:15  ACL MTU: 1021:9  SCO MTU: 255:4
        UP RUNNING
        RX bytes:1352 acl:0 sco:0 events:74 errors:0
        TX bytes:870 acl:0 sco:0 commands:74 errors:0
        Features: 0xbf 0xee 0xcd 0xfe 0xd8 0x3f 0x7b 0x87
        Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV3
        Link policy: RSWITCH SNIFF
        Link mode: SLAVE ACCEPT
        Name: ''
        Class: 0x000000
        Service Classes: Unspecified
        Device Class: Miscellaneous,
        HCI Version: 5.2 (0xb)  Revision: 0x14
        LMP Version: 5.2 (0xb)  Subversion: 0x14
        Manufacturer: RivieraWaves S.A.S (96)

root@TinaLinux:/# bluetoothd -n -d &              //启动蓝牙协议栈
root@TinaLinux:/# bluetoothd[645]: Bluetooth daemon 5.54
bluetoothd[645]: src/main.c:parse_config() parsing /etc/bluetooth/main.conf
bluetoothd[645]: src/main.c:parse_config() Key file does not have key 'DiscoverableTimeout'
     in group 'General'
bluetoothd[645]: src/main.c:parse_config() Key file does not have key 'AlwaysPairable' in
    group 'General'
bluetoothd[645]: src/main.c:parse_config() Key file does not have key 'PairableTimeout' in
    group 'General'
bluetoothd[645]: src/main.c:parse_config() Key file does not have key 'Privacy' in group '
    General'
bluetoothd[645]: src/main.c:parse_config() Key file does not have key 'JustWorksRepairing'
    in group 'General'
bluetoothd[645]: src/main.c:parse_config() Key file does not have key 'Name' in group '
    General'
bluetoothd[645]: src/main.c:parse_config() Key file does not have key 'Class' in group '
    General'
bluetoothd[645]: src/main.c:parse_config() Key file does not have key 'DeviceID' in group '
    General'
bluetoothd[645]: src/main.c:parse_config() Key file does not have key '
    ReverseServiceDiscovery' in group 'General'
bluetoothd[645]: src/main.c:parse_config() Key file does not have key 'Cache' in group '
    GATT'
bluetoothd[645]: src/main.c:parse_config() Key file does not have key 'KeySize' in group '
    GATT'
bluetoothd[645]: src/main.c:parse_config() Key file does not have key 'ExchangeMTU' in
    group 'GATT'
```

```
bluetoothd[645]: src/main.c:parse_config() Key file does not have key 'Channels' in group '
    GATT'
bluetoothd[645]: src/adapter.c:adapter_init() sending read version command
bluetoothd[645]: Starting SDP server
bluetoothd[645]: src/sdpd-service.c:register_device_id() Adding device id record for 0002:1
    d6b:0246:0536
bluetoothd[645]: src/plugin.c:plugin_init() Loading builtin plugins
bluetoothd[645]: src/plugin.c:add_plugin() Loading hostname plugin
bluetoothd[645]: src/plugin.c:add_plugin() Loading wiimote plugin
bluetoothd[645]: src/plugin.c:add_plugin() Loading autopair plugin
bluetoothd[645]: src/plugin.c:add_plugin() Loading policy plugin
bluetoothd[645]: src/plugin.c:add_plugin() Loading a2dp plugin
bluetoothd[645]: src/plugin.c:add_plugin() Loading avrcp plugin
bluetoothd[645]: src/plugin.c:add_plugin() Loading network plugin
bluetoothd[645]: src/plugin.c:add_plugin() Loading input plugin
bluetoothd[645]: src/plugin.c:add_plugin() Loading hog plugin
bluetoothd[645]: src/plugin.c:add_plugin() Loading gap plugin
bluetoothd[645]: src/plugin.c:add_plugin() Loading scanparam plugin
bluetoothd[645]: src/plugin.c:add_plugin() Loading deviceinfo plugin
bluetoothd[645]: src/plugin.c:add_plugin() Loading battery plugin
bluetoothd[645]: src/plugin.c:plugin_init() Loading plugins /usr/lib/bluetooth/plugins
bluetoothd[645]: Failed to init battery plugin
bluetoothd[645]: profiles/input/suspend-none.c:suspend_init()
bluetoothd[645]: profiles/network/manager.c:read_config() /etc/bluetooth/network.conf: Key
    file does not have key 'DisableSecurity' in group 'General'
bluetoothd[645]: profiles/network/manager.c:read_config() Config options: Security=true
bluetoothd[645]: kernel lacks bnep-protocol support
bluetoothd[645]: System does not support network plugin
bluetoothd[645]: src/main.c:main() Entering main loop
bluetoothd[645]: src/rfkill.c:rfkill_event() RFKILL event idx 0 type 2 op 0 soft 0 hard 0
bluetoothd[645]: Bluetooth management interface 1.14 initialized
bluetoothd[645]: src/adapter.c:read_version_complete() sending read supported commands
    command
bluetoothd[645]: src/adapter.c:read_version_complete() sending read index list command
bluetoothd[645]: src/rfkill.c:rfkill_event() RFKILL event idx 1 type 2 op 0 soft 1 hard 0
bluetoothd[645]: src/rfkill.c:rfkill_event() RFKILL event idx 2 type 1 op 0 soft 0 hard 0
bluetoothd[645]: src/adapter.c:read_commands_complete() Number of commands: 65
bluetoothd[645]: src/adapter.c:read_commands_complete() Number of events: 35
bluetoothd[645]: src/adapter.c:read_commands_complete() enabling kernel-side connection
    control
bluetoothd[645]: src/rfkill.c:rfkill_event() RFKILL event idx 3 type 2 op 0 soft 0 hard 0
bluetoothd[645]: src/adapter.c:read_index_list_complete() Number of controllers: 1
bluetoothd[645]: src/adapter.c:read_index_list_complete() Found index 0
bluetoothd[645]: src/adapter.c:index_added() index 0
bluetoothd[645]: src/adapter.c:btd_adapter_new() System name: BlueZ 5.54
bluetoothd[645]: src/adapter.c:btd_adapter_new() Major class: 0
bluetoothd[645]: src/adapter.c:btd_adapter_new() Minor class: 0
bluetoothd[645]: src/adapter.c:btd_adapter_new() Modalias: usb:v1D6Bp0246d0536
bluetoothd[645]: src/adapter.c:btd_adapter_new() Discoverable timeout: 180 seconds
bluetoothd[645]: src/adapter.c:btd_adapter_new() Pairable timeout: 0 seconds
bluetoothd[645]: src/adapter.c:index_added() sending read info command for index 0
bluetoothd[645]: src/adapter.c:read_info_complete() index 0 status 0x00
bluetoothd[645]: src/adapter.c:clear_uuids() sending clear uuids command for index 0
bluetoothd[645]: src/adapter.c:clear_devices() sending clear devices command for index 0
bluetoothd[645]: src/adapter.c:set_mode() sending set mode command for index 0
bluetoothd[645]: src/adapter.c:set_mode() sending set mode command for index 0
bluetoothd[645]: src/adapter.c:set_mode() sending set mode command for index 0
bluetoothd[645]: src/adapter.c:set_privacy() sending set privacy command for index 0
bluetoothd[645]: src/adapter.c:set_privacy() setting privacy mode 0x00 for index 0
bluetoothd[645]: src/advertising.c:btd_adv_manager_new() LE Advertising Manager created for
```

```
    adapter: /org/bluez/hci0
bluetoothd[645]: profiles/audio/a2dp.c:media_server_probe() path /org/bluez/hci0
bluetoothd[645]: plugins/hostname.c:hostname_probe()
bluetoothd[645]: profiles/audio/avrcp.c:avrcp_controller_server_probe() path /org/bluez/
    hci0
bluetoothd[645]: src/adapter.c:adapter_service_add() /org/bluez/hci0
bluetoothd[645]: src/sdpd-service.c:add_record_to_server() Adding record with handle 0
    x10001
bluetoothd[645]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00000017-0000-1000-8000-00805f9
bluetoothd[645]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00000100-0000-1000-8000-00805f9
bluetoothd[645]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00001002-0000-1000-8000-00805f9
bluetoothd[645]: src/sdpd-service.c:add_record_to_server() Record pattern UUID 0000110e
    -0000-1000-8000-00805f9
bluetoothd[645]: src/sdpd-service.c:add_record_to_server() Record pattern UUID 0000110f
    -0000-1000-8000-00805f9
bluetoothd[645]: src/adapter.c:adapter_service_insert() /org/bluez/hci0
bluetoothd[645]: src/adapter.c:add_uuid() sending add uuid command for index 0
bluetoothd[645]: profiles/audio/avrcp.c:avrcp_target_server_probe() path /org/bluez/hci0
bluetoothd[645]: src/adapter.c:adapter_service_add() /org/bluez/hci0
bluetoothd[645]: src/sdpd-service.c:add_record_to_server() Adding record with handle 0
    x10002
bluetoothd[645]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00000017-0000-1000-8000-00805f9
bluetoothd[645]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00000100-0000-1000-8000-00805f9
bluetoothd[645]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00001002-0000-1000-8000-00805f9
bluetoothd[645]: src/sdpd-service.c:add_record_to_server() Record pattern UUID 0000110c
    -0000-1000-8000-00805f9
bluetoothd[645]: src/sdpd-service.c:add_record_to_server() Record pattern UUID 0000110e
    -0000-1000-8000-00805f9
bluetoothd[645]: src/adapter.c:adapter_service_insert() /org/bluez/hci0
bluetoothd[645]: src/adapter.c:add_uuid() sending add uuid command for index 0
bluetoothd[645]: profiles/audio/a2dp.c:a2dp_sink_server_probe() path /org/bluez/hci0
bluetoothd[645]: profiles/audio/a2dp.c:a2dp_source_server_probe() path /org/bluez/hci0
bluetoothd[645]: src/adapter.c:btd_adapter_unblock_address() hci0 00:00:00:00:00:00
bluetoothd[645]: src/adapter.c:load_link_keys() hci0 keys 0 debug_keys 0
bluetoothd[645]: src/adapter.c:load_ltks() hci0 keys 0
bluetoothd[645]: src/adapter.c:load_irks() hci0 irks 0
bluetoothd[645]: src/adapter.c:load_conn_params() hci0 conn params 0
bluetoothd[645]: src/adapter.c:load_connections() sending get connections command for index
     0
bluetoothd[645]: src/adapter.c:adapter_service_insert() /org/bluez/hci0
bluetoothd[645]: src/adapter.c:add_uuid() sending add uuid command for index 0
bluetoothd[645]: src/adapter.c:set_did() hci0 source 2 vendor 1d6b product 246 version 536
bluetoothd[645]: src/adapter.c:adapter_register() Adapter /org/bluez/hci0 registered
bluetoothd[645]: src/adapter.c:set_dev_class() sending set device class command for index 0
bluetoothd[645]: src/adapter.c:set_name() sending set local name command for index 0
bluetoothd[645]: src/adapter.c:adapter_start() adapter /org/bluez/hci0 has been enabled
bluetoothd[645]: src/adapter.c:new_settings_callback() Settings: 0x000000c1
bluetoothd[645]: src/adapter.c:settings_changed() Changed settings: 0x00000040
bluetoothd[645]: src/adapter.c:settings_changed() Pending settings: 0x00000000
bluetoothd[645]: src/adapter.c:new_settings_callback() Settings: 0x000002c1
bluetoothd[645]: src/adapter.c:settings_changed() Changed settings: 0x00000200
bluetoothd[645]: src/adapter.c:settings_changed() Pending settings: 0x00000000
bluetoothd[645]: src/adapter.c:trigger_passive_scanning()
bluetoothd[645]: src/adapter.c:new_settings_callback() Settings: 0x00000ac1
```

```
bluetoothd[645]: src/adapter.c:settings_changed() Changed settings: 0x00000800
bluetoothd[645]: src/adapter.c:settings_changed() Pending settings: 0x00000000
bluetoothd[645]: Failed to set privacy: Rejected (0x0b)
bluetoothd[645]: src/adapter.c:load_link_keys_complete() link keys loaded for hci0
bluetoothd[645]: src/adapter.c:load_ltks_complete() LTKs loaded for hci0
bluetoothd[645]: src/adapter.c:load_irks_complete() IRKs loaded for hci0
bluetoothd[645]: src/adapter.c:load_conn_params_complete() Connection Parameters loaded for
     hci0
bluetoothd[645]: src/adapter.c:get_connections_complete() Connection count: 0
bluetoothd[645]: src/adapter.c:local_name_changed_callback() Name: BlueZ 5.54
bluetoothd[645]: src/adapter.c:local_name_changed_callback() Short name:
bluetoothd[645]: src/adapter.c:local_name_changed_callback() Current alias: BlueZ 5.54

root@TinaLinux:/# hcitool -i hci0 scan         //扫描周围蓝牙设备
Scanning ...
        A4:4B:D5:72:8C:9A       Redmi
        E0:DC:FF:E9:34:1E       flyBT
        B8:94:36:DD:8F:5B       jianjun_2.4G
```

5.1.6.2 使用 demo 测试验证

使用 demo 测试之前要修改 bt_init.sh 脚本，使用 AW869B 方案的指令。路径为/tina/target/allwinner/d1-nezha/base-files/etc/bluetooth/bt_init.sh。

```
#!/bin/sh
bt_hciattach="hciattach"
start_hci_attach()
{
    h=`ps | grep "$bt_hciattach" | grep -v grep`
    [ -n "$h" ] && {
        killall "$bt_hciattach"
        sleep 1
    }

    #aic h5 init
    echo 0 > /sys/class/rfkill/rfkill0/state;
    sleep 1
    echo 1 > /sys/class/rfkill/rfkill0/state;
    sleep 1

    "$bt_hciattach" -n -s 1500000 /dev/ttyS1 aic >/dev/null 2>&1 &
    sleep 1

    wait_hci0_count=0
    while true
    do
        [ -d /sys/class/bluetooth/hci0 ] && break
        sleep 1
        let wait_hci0_count++
        [ $wait_hci0_count -eq 8 ] && {
            echo "bring up hci0 failed"
            exit 1
        }
    done
}

start() {
```

```
    if [ -d "/sys/class/bluetooth/hci0" ];then
        echo "Bluetooth init has been completed!!"
    else
        start_hci_attach
    fi

    d=`ps | grep bluetoothd | grep -v grep`
    [ -z "$d" ] && {
#       bluetoothd -n  &
        /etc/bluetooth/bluetoothd start
        sleep 1
    }

}

ble_start() {
    if [ -d "/sys/class/bluetooth/hci0" ];then
        echo "Bluetooth init has been completed!!"
    else
        start_hci_attach
    fi

    hci_is_up=`hciconfig hci0 | grep RUNNING`
    [ -z "$hci_is_up" ] && {
        hciconfig hci0 up
    }

    MAC_STR=`hciconfig | grep "BD Address" | awk '{print $3}'`
    LE_MAC=${MAC_STR/2/C}
    OLD_LE_MAC_T=`cat /sys/kernel/debug/bluetooth/hci0/random_address`
    OLD_LE_MAC=$(echo $OLD_LE_MAC_T | tr [a-z] [A-Z])
    if [ -n "$LE_MAC" ];then
        if [ "$LE_MAC" != "$OLD_LE_MAC" ];then
            hciconfig hci0 lerandaddr $LE_MAC
        else
            echo "the ble random_address has been set."
        fi
    fi
}

stop() {
    d=`ps | grep bluetoothd | grep -v grep`
    [ -n "$d" ] && {
        killall bluetoothd
    }

    h=`ps | grep "$bt_hciattach" | grep -v grep`
    [ -n "$h" ] && {
        killall "$bt_hciattach"
        sleep 1
    }
    echo 0 > /sys/class/rfkill/rfkill0/state;
    echo "stop bluetoothd and hciattach"
}

case "$1" in
  start|"")
        start
        ;;
```

```
  stop)
        stop
        ;;
  ble_start)
        ble_start
        ;;
  *)
        echo "Usage: $0 {start|stop}"
        exit 1
esac
```

再次编译打包固件，然后启动 demo 的 bt_test 进行测试

```
root@TinaLinux:/# bt_test -i -d4
```

用手机连接蓝牙，播放音乐验证 BT 是否正常。

暂不支持 BLE 验证。

如遇到异常问题，请开发者参考最后一章**常见问题排查指南**分析。

5.2 XR819S 模组移植

```
主控：R528
无线模组：XR819s
内核版本：linux-5.4
方案：r528-evb2
```

5.2.1 确认硬件的工作条件配置软件输出

该部分的硬件条件需要关心的有：电源、复位、休眠唤醒、UART、主次时钟。然后对照硬件原理图，确认相应的引脚。xr819s 的原理图引脚图如下所示。

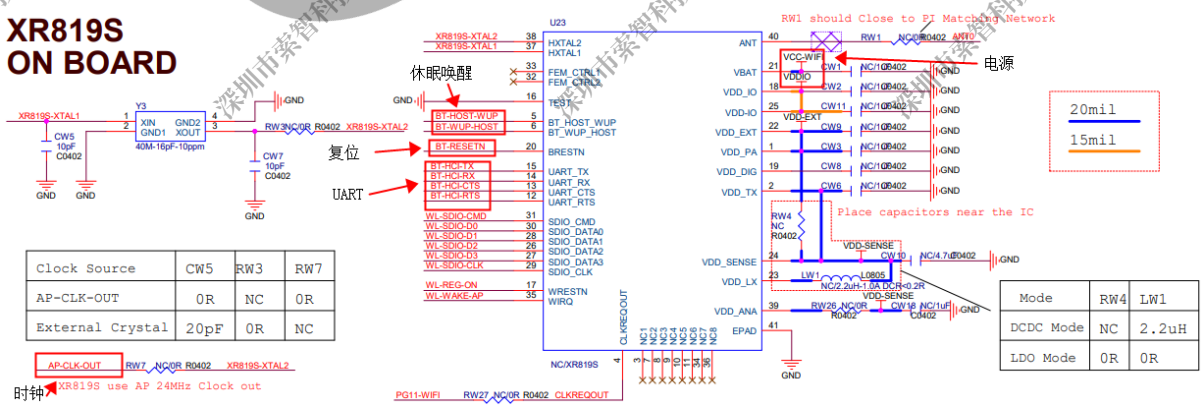

图 5-5: XR819s 原理图

5.2.1.1 电源

根据原理图找到 XR819s 的 VCC-WIFI 和 VDDIO 的供电方式 DC/DC5，属于默认开启的电源，不需要额外配置。

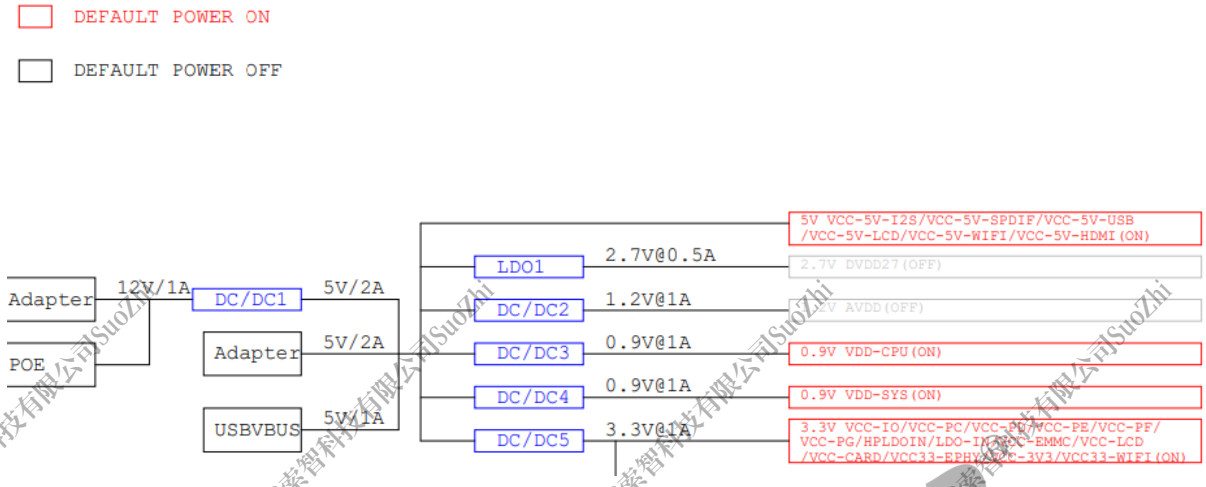

图 5-6: R528EVB2 电源树

5.2.1.2 复位使能

根据原理图中找到 BT-RESETN 连接的是 GPIO PG18，因此设备树中配置 bt_rst_n 标识为 PG18。配置文件路径是 tina/device/config/chips/r528/configs/evb2/board.dts

```
bt: bt@0 {
    compatible    = "allwinner,sunxi-bt";
    clock-names = "32k-fanout1";
    clocks = <&ccu CLK_FANOUT1_OUT>;
    /*bt_power_num = <0x01>;*/
    /*bt_power      = "axp803-dldo1";*/
    /*bt_io_regulator = "axp803-dldo1";*/
    /*bt_io_vol = <3300000>;*/
    /*bt_power_vol = <330000>;*/
    bt_rst_n     = <&pio PG 18 GPIO_ACTIVE_LOW>;   //复位引脚配置
    status        = "okay";
};
```

5.2.1.3 休眠唤醒

参考确认硬件的工作条件配置软件输出-> 休眠唤醒功能 >，根据原理图中找到 bt_wake 和 bt_hostwake 分别对应 GPIO PG16、GPIO PG17，注意配置 dts 时 bt_wake 默认输出高平。

配置文件路径是 tina/device/config/chips/r528/configs/evb2/board.dts

```
btlpm: btlpm@0 {
    compatible  = "allwinner,sunxi-btlpm";
    uart_index  = <0x1>;
    bt_wake     = <&pio PG 16 GPIO_ACTIVE_HIGH>;   // bt_wake的引脚配置
    bt_hostwake = <&pio PG 17 GPIO_ACTIVE_HIGH>;   // bt_hostwake 的引脚配置
    status      = "okay";
};
```

5.2.1.4 UART 接口配置

根据原理图中找到 UART 引脚 PG06 PG07 PG08 PG09，在设备树里进行配置并使能。

配置文件路径是 tina/device/config/chips/r528/configs/evb2/board.dts

```
uart1_pins_a: uart1_pins@0 {  /* For EVB2 board */
    pins = "PG6", "PG7", "PG8", "PG9";
    function = "uart1";
    drive-strength = <10>;
    bias-pull-up;
};

uart1_pins_b: uart1_pins@1{  /* For EVB2 board */
    pins = "PG6", "PG7", "PG8", "PG9";
    function = "gpio_in";
};

&uart1 {
    pinctrl-names = "default", "sleep";
    pinctrl-0 = <&uart1_pins_a>;
    pinctrl-1 = <&uart1_pins_b>;
    status = "okay";
};
```

5.2.1.5 主次时钟

根据原理图看出主时钟由外部晶振提供，次时钟是主控输出。需要配置主控端的 CLK 驱动。

配置文件路径是 tina/device/config/chips/r528/configs/evb2/board.dts

```
bt: bt@0 {
    compatible    = "allwinner,sunxi-bt";
    clock-names = "32k-fanout1";
    clocks = <&ccu CLK_FANOUT1_OUT>;   //时钟配置
    /*bt_power_num = <0x01>;*/
    /*bt_power      = "axp803-dldo1";*/
    /*bt_io_regulator = "axp803-dldo1";*/
    /*bt_io_vol = <3300000>;*/
    /*bt_power_vol = <330000>;*/
    bt_rst_n      = <&pio PG 18 GPIO_ACTIVE_LOW>;
    status        = "okay";
};
```

5.2.2 sunxi-rf 驱动

使能 sunx-rf 驱动，用于解析上一章节中 dts 中的配置项

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

5.2.3 使能蓝牙协议栈

（1）协议栈内核空间的配置

```
make kernel_menuconfig
[*] Networking support  --->
  <*>   Bluetooth subsystem support  --->
    [ ]   Bluetooth Classic (BR/EDR) features //经典蓝牙特性支持
    [*]   Bluetooth Low Energy (LE) features   //低功耗蓝牙特性支持
    [ ]   Enable LED triggers
    [ ]   Bluetooth self testing support
    [*]   Export Bluetooth internals in debugfs   //蓝牙调试节点
          Bluetooth device drivers  --->
      <*> HCI UART driver
      [*]   UART (H4) protocol support    //hci uart h4驱动
      <*> Xradio Bluetooth sleep driver support   //lpm驱动
```

（2）协议栈用户空间的配置

```
make menuconfig
Firmware  --->
  -*- xr819s with 40M sdd
Kernel modules  --->
  Wireless Drivers  --->
    <*> kmod-net-xr819s-40M
Utilities  --->
  -*- bluez-daemon
  -*- bluez-utils          //协议栈调试工具
  -*- bluez-utils-extra
  rf test tool --->
    <*> xr819s-rftest       //rf测试工具
```

5.2.4 移植 hciattach

XR819s 的 hciattach 是基于原生 bluez 协议栈适配的，是在原 XR829 的 hciattach_xradio.c 上添加了 XR819S 部分。可以参考 tina/package/utils/bluez/patches/011-add-xr819s-hciattach-support.patch。

编译后生成的 hciattach.c 可以在路径在 tina/out/r528-evb2/compile_dir/target/bluez-5.54/tools/里面看到。

5.2.5 调试验证

配置完编译打包生成固件烧录到设备中，准备测试验证。819s 只有 BLE，没有经典蓝牙，所以只需验证 BLE 部分即可。

5.2.5.1 分步骤测试验证

```
1. 硬件工作条件：供电使能
echo 0 > /sys/class/rfkill/rfkill0/state //蓝牙复位
echo 1 > /sys/class/rfkill/rfkill0/state //蓝牙上电使能
2. 蓝牙初始化
hciattach -n ttyS1 xr819s & //以xr819s为例，不同的模组厂商，命令不一样。
3. 启动协议栈
bluetoothd -n -d &
4. 网卡启动
hciconfig hci0 up
5. 验证功能
hciconfig -a //查询状态
hcitool -i hci0 lescan //扫描
```

按照上面的指令在终端执行 log 如下：

```
root@TinaLinux:/# hciattach -n ttyS1 xr819s &    //蓝牙初始化
root@TinaLinux:/# xradio_init
[  274.013804] sunxi-rfkill soc@3000000:rfkill@0: set block: 1
[  274.020908] sunxi-rfkill soc@3000000:rfkill@0: bt power off success
[  274.528571] sunxi-rfkill soc@3000000:rfkill@0: set block: 0
[  274.544864] sunxi-rfkill soc@3000000:rfkill@0: bt power on success
[  274.572339] [XR_BT_LPM] bluedroid_write_proc_btwake: bluedroid_write_proc_btwake 1
[  274.580852] [XR_BT_LPM] bluedroid_write_proc_btwake: wakeup bt device
[  274.588161] [XR_BT_LPM] bluedroid_write_proc_lpm: disable lpm mode
set LPM mode:disabled[userial_sync] uart sync count: 1.
[userial_sync] read buf: 00 00.
[userial_sync] uart sync count: 2.
[userial_sync] read buf: 4f 4b.
[userial_sync] Receive OK, uart sync done.
Set uart mode done
[userial_sync] uart sync count: 1.
[userial_sync] read buf: 00 00.
[userial_sync] uart sync count: 2.
[userial_sync] read buf: 4f 4b.
[userial_sync] Receive OK, uart sync done.
[load_btfirmware] start loading firmware...
[load_btfirmware] open firmware file success. loading...
remain len:     0x2076C.
remain len:     0x2036C.
remain len:     0x1FF6C.
...
...
remain len:     0x076C.
remain len:     0x036C.
remain len:     0x0000.
load firmware done.
jump:
set pc 201101, val 1112000
```

```
Now the system will jump to 00201101
Set HW FlowControl Off
userial_vendor_set_hw_fctrl set hw flowcontrol off
[xradio_init] send reset cmd...
writing
01 03 0c 00
received 7
04 0e 04 01 03 0c 00
[xradio_init] set bdaddr...
writing
01 0a fc 09 02 00 06 7f 9d 8e de 22 22
received 7
04 0e 04 01 0a fc 00
writing
01 03 0c 00
received 7
04 0e 04 01 03 0c 00
[xradio_init] bring up hci...
Done setting line discpline
Device setup complete
//回车
root@TinaLinux:/# bluetoothd -n -d &      //启动蓝牙协议栈
root@TinaLinux:/# bluetoothd[1790]: Bluetooth daemon 5.54
bluetoothd[1790]: src/main.c:parse_config() parsing /etc/bluetooth/main.conf
bluetoothd[1790]: src/main.c:parse_config() Key file does not have key 'DiscoverableTimeout
    ' in group 'General'
bluetoothd[1790]: src/main.c:parse_config() Key file does not have key 'AlwaysPairable' in
    group 'General'
bluetoothd[1790]: src/main.c:parse_config() Key file does not have key 'PairableTimeout' in
     group 'General'
bluetoothd[1790]: src/main.c:parse_config() Key file does not have key 'Privacy' in group '
    General'
bluetoothd[1790]: src/main.c:parse_config() Key file does not have key 'JustWorksRepairing'
     in group 'General'
bluetoothd[1790]: src/main.c:parse_config() Key file does not have key 'Name' in group '
    General'
bluetoothd[1790]: src/main.c:parse_config() Key file does not have key 'Class' in group '
    General'
bluetoothd[1790]: src/main.c:parse_config() Key file does not have key 'DeviceID' in group
    'General'
bluetoothd[1790]: src/main.c:parse_config() Key file does not have key '
    ReverseServiceDiscovery' in group 'General'
bluetoothd[1790]: src/main.c:parse_config() Key file does not have key 'Cache' in group '
    GATT'
bluetoothd[1790]: src/main.c:parse_config() Key file does not have key 'KeySize' in group '
    GATT'
bluetoothd[1790]: src/main.c:parse_config() Key file does not have key 'ExchangeMTU' in
    group 'GATT'
bluetoothd[1790]: src/main.c:parse_config() Key file does not have key 'Channels' in group
    'GATT'
bluetoothd[1790]: src/adapter.c:adapter_init() sending read version command
bluetoothd[1790]: Starting SDP server
bluetoothd[1790]: src/sdpd-service.c:register_device_id() Adding device id record for
    0002:1d6b:0246:0536
bluetoothd[1790]: src/plugin.c:plugin_init() Loading builtin plugins
bluetoothd[1790]: src/plugin.c:add_plugin() Loading hostname plugin
bluetoothd[1790]: src/plugin.c:add_plugin() Loading wiimote plugin
bluetoothd[1790]: src/plugin.c:add_plugin() Loading autopair plugin
bluetoothd[1790]: src/plugin.c:add_plugin() Loading policy plugin
bluetoothd[1790]: src/plugin.c:add_plugin() Loading a2dp plugin
```

```
bluetoothd[1790]: src/plugin.c:add_plugin() Loading avrcp plugin
bluetoothd[1790]: src/plugin.c:add_plugin() Loading network plugin
bluetoothd[1790]: src/plugin.c:add_plugin() Loading input plugin
bluetoothd[1790]: src/plugin.c:add_plugin() Loading hog plugin
bluetoothd[1790]: src/plugin.c:add_plugin() Loading gap plugin
bluetoothd[1790]: src/plugin.c:add_plugin() Loading scanparam plugin
bluetoothd[1790]: src/plugin.c:add_plugin() Loading deviceinfo plugin
bluetoothd[1790]: src/plugin.c:add_plugin() Loading battery plugin
bluetoothd[1790]: src/plugin.c:plugin_init() Loading plugins /usr/lib/bluetooth/plugins
bluetoothd[1790]: Failed to init battery plugin
bluetoothd[1790]: profiles/input/suspend-none.c:suspend_init()
bluetoothd[1790]: profiles/network/manager.c:read_config() /etc/bluetooth/network.conf: Key
     file does not have key 'DisableSecurity' in group 'General'
bluetoothd[1790]: profiles/network/manager.c:read_config() Config options: Security=true
bluetoothd[1790]: kernel lacks bnep-protocol support
bluetoothd[1790]: System does not support network plugin
bluetoothd[1790]: src/main.c:main() Entering main loop
bluetoothd[1790]: src/rfkill.c:rfkill_event() RFKILL event idx 0 type 2 op 0 soft 0 hard 0
bluetoothd[1790]: Bluetooth management interface 1.14 initialized
bluetoothd[1790]: src/adapter.c:read_version_complete() sending read supported commands
    command
bluetoothd[1790]: src/adapter.c:read_version_complete() sending read index list command
bluetoothd[1790]: src/rfkill.c:rfkill_event() RFKILL event idx 1 type 1 op 0 soft 0 hard 0
bluetoothd[1790]: src/rfkill.c:rfkill_event() RFKILL event idx 2 type 2 op 0 soft 0 hard 0
bluetoothd[1790]: src/adapter.c:read_commands_complete() Number of commands: 65
bluetoothd[1790]: src/adapter.c:read_commands_complete() Number of events: 35
bluetoothd[1790]: src/adapter.c:read_commands_complete() enabling kernel-side connection
    control
bluetoothd[1790]: src/adapter.c:read_index_list_complete() Number of controllers: 1
bluetoothd[1790]: src/adapter.c:read_index_list_complete() Found index 0
bluetoothd[1790]: src/adapter.c:index_added() index 0
bluetoothd[1790]: src/adapter.c:btd_adapter_new() System name: BlueZ 5.54
bluetoothd[1790]: src/adapter.c:btd_adapter_new() Major class: 0
bluetoothd[1790]: src/adapter.c:btd_adapter_new() Minor class: 0
bluetoothd[1790]: src/adapter.c:btd_adapter_new() Modalias: usb:v1D6Bp0246d0536
bluetoothd[1790]: src/adapter.c:btd_adapter_new() Discoverable timeout: 180 seconds
bluetoothd[1790]: src/adapter.c:btd_adapter_new() Pairable timeout: 0 seconds
bluetoothd[1790]: src/adapter.c:index_added() sending read info command for index 0
bluetoothd[1790]: src/adapter.c:read_info_complete() index 0 status 0x00
bluetoothd[1790]: src/adapter.c:clear_uuids() sending clear uuids command for index 0
bluetoothd[1790]: src/adapter.c:clear_devices() sending clear devices command for index 0
bluetoothd[1790]: src/adapter.c:set_mode() sending set mode command for index 0
bluetoothd[1790]: src/adapter.c:set_privacy() sending set privacy command for index 0
bluetoothd[1790]: src/adapter.c:set_privacy() setting privacy mode 0x00 for index 0
bluetoothd[1790]: src/advertising.c:btd_adv_manager_new() LE Advertising Manager created
    for adapter: /org/bluez/hci0
bluetoothd[1790]: profiles/audio/a2dp.c:media_server_probe() path /org/bluez/hci0
bluetoothd[1790]: plugins/hostname.c:hostname_probe()
bluetoothd[1790]: profiles/audio/avrcp.c:avrcp_controller_server_probe() path /org/bluez/
    hci0
bluetoothd[1790]: src/adapter.c:adapter_service_add() /org/bluez/hci0
bluetoothd[1790]: src/sdpd-service.c:add_record_to_server() Adding record with handle 0
    x10001
bluetoothd[1790]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00000017-0000-1000-8000-00805f9
bluetoothd[1790]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00000100-0000-1000-8000-00805f9
bluetoothd[1790]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00001002-0000-1000-8000-00805f9
bluetoothd[1790]: src/sdpd-service.c:add_record_to_server() Record pattern UUID 0000110e
```

```
    -0000-1000-8000-00805f9
bluetoothd[1790]: src/sdpd-service.c:add_record_to_server() Record pattern UUID 0000110f
    -0000-1000-8000-00805f9
bluetoothd[1790]: src/adapter.c:adapter_service_insert() /org/bluez/hci0
bluetoothd[1790]: src/adapter.c:add_uuid() sending add uuid command for index 0
bluetoothd[1790]: profiles/audio/avrcp.c:avrcp_target_server_probe() path /org/bluez/hci0
bluetoothd[1790]: src/adapter.c:adapter_service_add() /org/bluez/hci0
bluetoothd[1790]: src/sdpd-service.c:add_record_to_server() Adding record with handle 0
    x10002
bluetoothd[1790]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00000017-0000-1000-8000-00805f9
bluetoothd[1790]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00000100-0000-1000-8000-00805f9
bluetoothd[1790]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00001002-0000-1000-8000-00805f9
bluetoothd[1790]: src/sdpd-service.c:add_record_to_server() Record pattern UUID 0000110c
    -0000-1000-8000-00805f9
bluetoothd[1790]: src/sdpd-service.c:add_record_to_server() Record pattern UUID 0000110e
    -0000-1000-8000-00805f9
bluetoothd[1790]: src/adapter.c:adapter_service_insert() /org/bluez/hci0
bluetoothd[1790]: src/adapter.c:add_uuid() sending add uuid command for index 0
bluetoothd[1790]: profiles/audio/a2dp.c:a2dp_sink_server_probe() path /org/bluez/hci0
bluetoothd[1790]: profiles/audio/a2dp.c:a2dp_source_server_probe() path /org/bluez/hci0
bluetoothd[1790]: src/adapter.c:btd_adapter_unblock_address() hci0 00:00:00:00:00:00
bluetoothd[1790]: src/adapter.c:load_ltks() hci0 keys 0
bluetoothd[1790]: src/adapter.c:load_irks() hci0 irks 0
bluetoothd[1790]: src/adapter.c:load_conn_params() hci0 conn params 0
bluetoothd[1790]: src/adapter.c:adapter_service_insert() /org/bluez/hci0
bluetoothd[1790]: src/adapter.c:add_uuid() sending add uuid command for index 0
bluetoothd[1790]: src/adapter.c:set_did() hci0 source 2 vendor 1d6b product 246 version 536
bluetoothd[1790]: src/adapter.c:adapter_register() Adapter /org/bluez/hci0 registered
bluetoothd[1790]: src/adapter.c:set_name() sending set local name command for index 0
bluetoothd[1790]: src/adapter.c:new_settings_callback() Settings: 0x00000a00
bluetoothd[1790]: src/adapter.c:settings_changed() Changed settings: 0x00000800
bluetoothd[1790]: src/adapter.c:settings_changed() Pending settings: 0x00000000
bluetoothd[1790]: src/adapter.c:set_privacy_complete() Successfuly set privacy for index 0
bluetoothd[1790]: src/adapter.c:load_ltks_complete() LTKs loaded for hci0
bluetoothd[1790]: src/adapter.c:load_irks_complete() IRKs loaded for hci0
bluetoothd[1790]: src/adapter.c:load_conn_params_complete() Connection Parameters loaded
    for hci0
bluetoothd[1790]: src/adapter.c:local_name_changed_callback() Name: BlueZ 5.54
bluetoothd[1790]: src/adapter.c:local_name_changed_callback() Short name:
bluetoothd[1790]: src/adapter.c:local_name_changed_callback() Current alias: BlueZ 5.54
//回车
root@TinaLinux:/# hciconfig hci0 up   //网卡启动
Can't init device hci0: Invalid argument (22)
bluetoothd[1790]: src/adapter.c:new_settings_callback() Settings: 0x00000a01
bluetoothd[1790]: src/adapter.c:settings_changed() Changed settings: 0x00000001
bluetoothd[1790]: src/adapter.c:settings_changed() Pending settings: 0x00000000
bluetoothd[1790]: src/adapter.c:adapter_start() adapter /org/bluez/hci0 has been enabled
bluetoothd[1790]: src/adapter.c:trigger_passive_scanning()
root@TinaLinux:/# hciconfig -a        //查看网卡状态
hci0:   Type: Primary  Bus: UART
        BD Address: 22:22:DE:8E:9D:7F  ACL MTU: 251:13  SCO MTU: 0:0
        UP RUNNING
        RX bytes:507 acl:0 sco:0 events:38 errors:0
        TX bytes:294 acl:0 sco:0 commands:38 errors:0
        Features: 0x00 0x00 0x00 0x00 0x60 0x00 0x00 0x00
        Packet type: DM1 DH1 HV1
        Link policy:
```

```
        Link mode: SLAVE ACCEPT
Can't read local name on hci0: Input/output error (5)
root@TinaLinux:/# hcitool -i hci0 lescan   //扫描周围设备
LE Scan ...
1B:E6:FD:74:47:7C (unknown)
7D:7C:57:DC:91:C7 (unknown)
32:C9:91:D3:A6:A1 (unknown)
FA:A5:B6:17:A5:A8 (unknown)
```

5.2.5.2 使用 demo 测试验证

测试之前要检查 bt_init.sh 脚本是否与方案对应。路径为 tina/target/allwinner/r328s2-std/base-files/etc/bluetooth/bt_init.sh。XR819s 使用的 xr819s_bt_init.sh 脚本。如果不对应，需要将目录/tina/target/allwinner/r528-evb2/base-files/etc/bluetooth/bt_init.sh 的脚本换成对应方案的脚本。

然后启动 demo 的 bt_gatt_server_test 进行测试

```
root@TinaLinux:/# bt_gatt_server_test
```

用手机 LightBlue 软件，看是否能连上设备释放出的热点，进行读写特性操作。

如遇到异常问题，请开发者参考最后一章**常见问题排查指南**分析。

6 模组从已支持平台添加到新平台示例

如果调试的物料发现已经在其他平台上有支持，那么就可以复用其他平台的成果，快速的达成移植过程。

6.1 XR829 模组移植

```
主控：R328
无线模组：XR829
内核版本：linux-4.9
方案：r328s2-std
```

6.1.1 确认硬件的工作条件配置软件输出

该部分的硬件条件需要关心的有：电源、复位、休眠唤醒、UART、主次时钟。然后对照硬件原理图，确认相应的引脚。XR829 的引脚图如下所示。

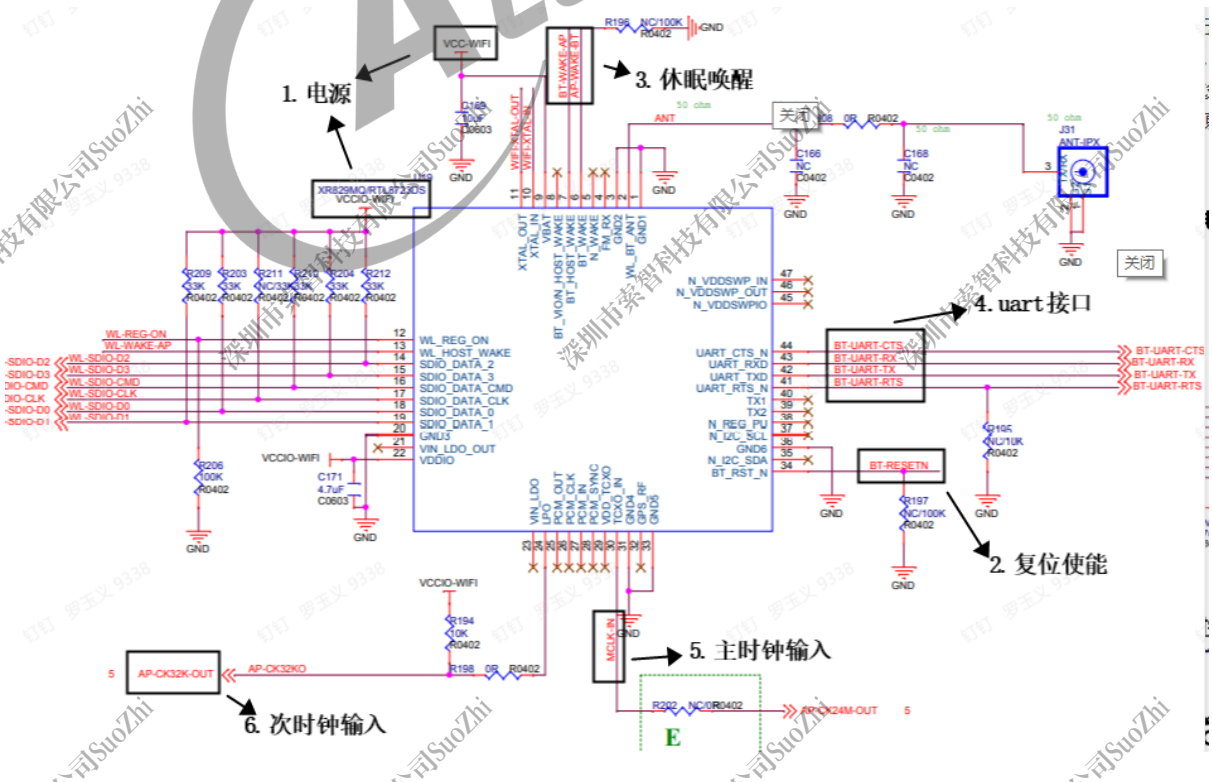

图 6-1: XR829 原理图

6.1.1.1 电源

XR829 是 WiFi 和 BT 合一的模组，供电部分是复用的。根据原理图找到 XR829 的 VCC-WIFI 和 VCCIO-WIFI 的供电方式是电源管理芯片，划分在 LDO3 电源域，从原理图中可以看到红色框表示 DEFAULT POWER ON，已经是默认输出 3.3V，所以不用再单独配。

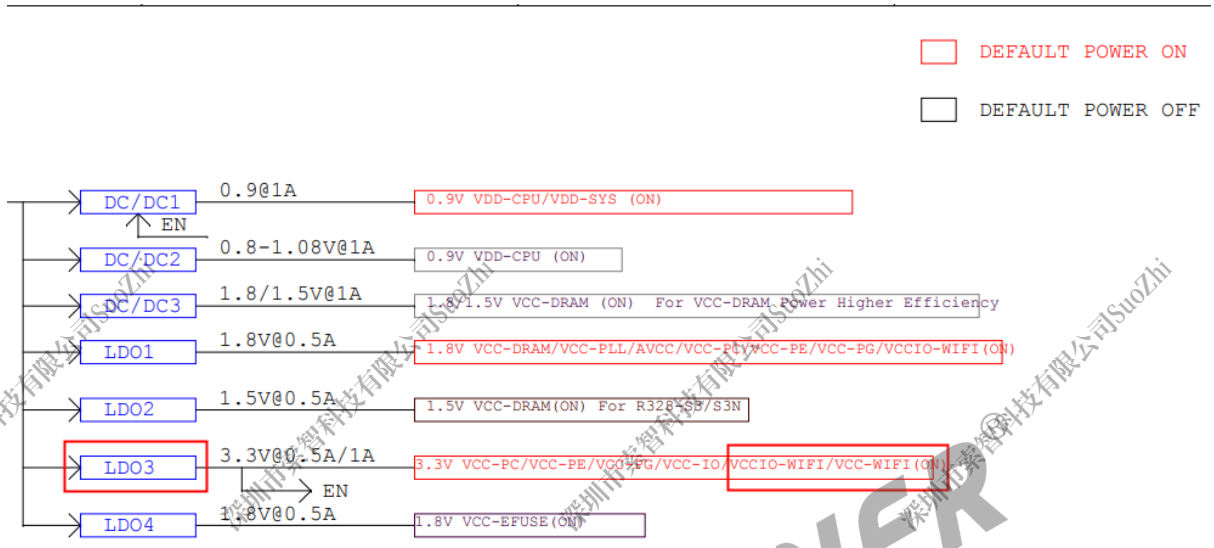

图 6-2: R328 电源树

6.1.1.2 复位使能

根据原理图中找到 BT-RESETN 连接的是 GPIO PE04，因此设备树中配置 bt_rst_n 标识为 PE04

配置文件路径是/tina/device/config/chips/r328s2/configs/std/sys_config.fex

```
[bt]
bt_used             = 1
compatible          = "allwinner,sunxi-bt"
clocks              = "losc_out"
bt_rst_n            = port:PE04<1><default><default><0>
```

图 6-3: 复位配置

6.1.1.3 休眠唤醒

上面**获取模组原厂提供的驱动资料包**章节中，已经告诉大家如何对 btlpm 进行配置，其次就是要在设备树里配置引脚，根据原理图中找到 bt_wake 和 bt_hostwake 分别对应 GPIO PE03、GPIO PE00，注意配置 dts 时 bt_wake 默认输出高平。

配置文件路径是/tina/device/config/chips/r328s2/configs/std/sys_config.fex

```
[btlpm]
btlpm_used          = 1
compatible          = "allwinner,sunxi-btlpm"
uart_index          = 1
bt_wake             = port:PE03<1><default><default><1>
bt_hostwake         = port:PE00<6><default><default><0>
```

图 6-4: lpm 引脚配置

6.1.1.4 UART 接口配置

根据原理图中找到 UART 引脚 PG06 PG07 PG08 PG09，在设备树里进行配置并使能。

配置文件路径是/tina/device/config/chips/r328s2/configs/std/sys_config.fex

```
[uart1]
uart1_used          = 1
uart1_type          = 4
uart1_tx            = port:PG06<2><1><default><default>
uart1_rx            = port:PG07<2><1><default><default>
uart1_rts           = port:PG08<2><1><default><default>
uart1_cts           = port:PG09<2><1><default><default>

[uart1_suspend]
uart1_tx            = port:PG06<7><1><default><default>
uart1_rx            = port:PG07<7><1><default><default>
uart1_rts           = port:PG08<7><1><default><default>
uart1_cts           = port:PG09<7><1><default><default>
```

图 6-5: UART1 串口配置

6.1.1.5 主时钟和次时钟

根据原理图看出主时钟由外部晶振提供，次时钟是主控输出。需要配置主控端的 CLK 驱动。

配置文件路径是/tina/device/config/chips/r328s2/configs/std/sys_config.fex

```
[bt]
bt_used                 = 1
compatible              = "allwinner,sunxi-bt"
clocks                  = "losc_out"
bt_rst_n                = port:PE04<1><default><default><0>

[btlpm]
btlpm_used              = 1
compatible              = "allwinner,sunxi-btlpm"
uart_index              = 1
bt_wake                 = port:PE03<1><default><default><1>
bt_hostwake             = port:PE00<6><default><default><0>
```

图 6-6: 时钟配置

6.1.2 内核配置和用户空间配置

通过界面配置的方式打开相应的选项，分为协议栈内核部分的配置和协议栈用户空间的配置。注意有些默认配置与本方案冲突的需取消勾选。下图的配置不仅展示了蓝牙协议栈部分，也包括 R328 适配 XR829 的其他重点配置：uart 驱动、fireware、kernel 模块、rf 测试工具。

(1) 协议栈内核空间的配置—>make kernel_menuconfig

```
make kernel_menuconfig
[*] Networking support  --->
    <*>   Bluetooth subsystem support  --->
        [*]   Bluetooth Classic (BR/EDR) features //经典蓝牙特性支持
        <*>     RFCOMM protocol support    //如果需要SPP文件传输等相关功能可以选上
        [*]       RFCOMM TTY support
        < >     BNEP protocol support
        < >     HIDP protocol support
        [ ]     Bluetooth High Speed (HS) features
        [*]   Bluetooth Low Energy (LE) features   //低功耗蓝牙特性支持
        [ ]   Enable LED triggers
        [ ]   Bluetooth self testing support
        [*]   Export Bluetooth internals in debugfs   //蓝牙调试节点
              Bluetooth device drivers  --->
          <*> Xradio Bluetooth sleep driver support   //lpm驱动
          <*> Xradio Bluetooth farmware debug interface support
          <*> HCI UART driver
          [*]   UART (H4) protocol support    //hci uart h4驱动
```

(2) 协议栈用户空间的配置—>make menuconfig

```
make menuconfig
 Firmware  --->
   <*> xr829-firmware
   [*] xr829 with 40M sdd
Kernel modules  --->
  Wireless Drivers  --->
    <*> kmod-net-xrbtlpm     //休眠唤醒功能
Utilities  --->
  -*- bluez-daemon
  [*] Enable xr829 extra config    //使能Xradio的额外配置
  [*]   Enable xr829 lpm   //使能休眠唤醒功能
  -*- bluez-utils          //协议栈调试工具
  -*- bluez-utils-extra
  rf test tool --->
    <*> xr829-rftest       //rf测试工具
```

6.1.3 调试验证

配置完编译打包生成固件烧录到设备中，准备测试验证。

6.1.3.1 分步骤测试验证

```
1. 硬件工作条件：供电使能
echo 0 > /sys/class/rfkill/rfkill0/state //蓝牙复位
echo 1 > /sys/class/rfkill/rfkill0/state //蓝牙上电使能
2. 蓝牙初始化
hciattach -n ttyS1 xradio & //以xr829为例，不同的模组厂商，命令不一样。
3. 启动协议栈
bluetoothd -n -d &
4. 网卡启动
hciconfig hci0 up
5. 验证功能
hciconfig -a   //查询状态
hcitool -i hci0 lescan  //扫描
```

按照上面的指令在终端执行 log 如下：

```
root@TinaLinux:/# echo 0 > /sys/class/rfkill/rfkill0/state   //蓝牙复位
root@TinaLinux:/# echo 1 > /sys/class/rfkill/rfkill0/state   //蓝牙上电使能
root@TinaLinux:/# hciattach -n ttyS1 xradio &    //蓝牙初始化
root@TinaLinux:/# xradio_init
set LPM mode:disabled[userial_sync] uart sync count: 1.
[userial_sync] read buf: 00 00.
[userial_sync] uart sync count: 2.
[userial_sync] read buf: 4f 4b.
[userial_sync] Receive OK, uart sync done.
Set uart mode done
[userial_sync] uart sync count: 1.
[userial_sync] read buf: 00 00.
[userial_sync] uart sync count: 2.
[userial_sync] read buf: 00 00.
[userial_sync] uart sync count: 3.
[userial_sync] read buf: 4f 4b.
[userial_sync] Receive OK, uart sync done.
```

```
[load_btfirmware] start loading firmware...
[load_btfirmware] open firmware file success.
load firmware done.
jump:
set pc 0, val 0
Now the system will jump to 00000000
Set HW FlowControl On
userial_vendor_set_hw_fctrl set hw flowcontrol on
[xradio_init] send reset cmd...
writing
01 03 0c 00
received 7
04 0e 04 05 03 0c 00
[xradio_init] update hci baudrate...
writing
01 18 fc 04 60 e3 16 00
received 7
04 0e 04 05 18 fc 00
Done setting baudrate
[xradio_init] set bdaddr...
bdaddr is null, not valid
writing
01 0a fc 09 02 00 06 2c 80 60 85 22 22
received 7
04 0e 04 05 0a fc 00
writing
01 03 0c 00
received 7
04 0e 04 05 03 0c 00
[xradio_init] bring up hci...
Done setting line discpline
Device setup complete
//回车
root@TinaLinux:/# bluetoothd -n -d &   //启动协议栈
root@TinaLinux:/# bluetoothd[1523]: Bluetooth daemon 5.54
bluetoothd[1523]: src/main.c:parse_config() parsing /etc/bluetooth/main.conf
bluetoothd[1523]: src/main.c:parse_config() Key file does not have key 'DiscoverableTimeout
    ' in group 'General'
bluetoothd[1523]: src/main.c:parse_config() Key file does not have key 'AlwaysPairable' in
    group 'General'
bluetoothd[1523]: src/main.c:parse_config() Key file does not have key 'PairableTimeout' in
     group 'General'
bluetoothd[1523]: src/main.c:parse_config() Key file does not have key 'Privacy' in group '
    General'
...
...
...
bluetoothd[1523]: src/adapter.c:new_settings_callback() Settings: 0x00000ac0
bluetoothd[1523]: src/adapter.c:settings_changed() Changed settings: 0x00000800
bluetoothd[1523]: src/adapter.c:settings_changed() Pending settings: 0x00000000
bluetoothd[1523]: src/adapter.c:set_privacy_complete() Successfuly set privacy for index 0
bluetoothd[1523]: src/adapter.c:add_whitelist_complete() 40:26:19:94:25:B0 added to kernel
    whitelist
bluetoothd[1523]: src/adapter.c:load_link_keys_complete() link keys loaded for hci0
bluetoothd[1523]: src/adapter.c:load_ltks_complete() LTKs loaded for hci0
bluetoothd[1523]: src/adapter.c:load_irks_complete() IRKs loaded for hci0
bluetoothd[1523]: src/adapter.c:load_conn_params_complete() Connection Parameters loaded
    for hci0
bluetoothd[1523]: src/adapter.c:local_name_changed_callback() Name: aw-bt-test-80-2C
bluetoothd[1523]: src/adapter.c:local_name_changed_callback() Short name:
```

```
bluetoothd[1523]: src/adapter.c:local_name_changed_callback() Current alias: aw-bt-test
    -80-2C

root@TinaLinux:/# hciconfig hci0 up    //网卡启动
bluetoothd[1523]: src/adapter.c:new_settings_callback() Settings: 0x00000ac1
bluetoothd[1523]: src/adapter.c:settings_changed() Changed settings: 0x00000001
bluetoothd[1523]: src/adapter.c:settings_changed() Pending settings: 0x00000000
bluetoothd[1523]: src/adapter.c:adapter_start() adapter /org/bluez/hci0 has been enabled
bluetoothd[1523]: src/adapter.c:trigger_passive_scanning()
root@TinaLinux:/# hciconfig -a        //查看网卡状态信息
hci0:   Type: Primary  Bus: UART
        BD Address: 22:22:85:60:80:2C  ACL MTU: 1021:8  SCO MTU: 255:4
        UP RUNNING PSCAN
        RX bytes:1182 acl:0 sco:0 events:58 errors:0
        TX bytes:1012 acl:0 sco:0 commands:58 errors:0
        Features: 0xbf 0xfe 0xcd 0xfe 0xdb 0xfd 0x7b 0x87
        Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
        Link policy: RSWITCH SNIFF
        Link mode: SLAVE ACCEPT
        Name: 'aw-bt-test-80-2C'
        Class: 0x000000
        Service Classes: Unspecified
        Device Class: Miscellaneous,
        HCI Version: 4.1 (0x7)  Revision: 0xc21
        LMP Version: 4.1 (0x7)  Subversion: 0xc21
        Manufacturer: not assigned (1597)

root@TinaLinux:/# hcitool -i hci0 lescan    //扫描周围设备
LE Scan ...
1E:08:2F:1C:70:E3 (unknown)
34:46:19:F7:D2:91 (unknown)
62:FA:11:FA:7C:5D (unknown)
DD:BD:04:79:92:8D (unknown)
DD:BD:04:79:92:8D XRADIOCAI
E2:C6:4A:47:D6:80 U-GTW945F
E2:C6:4A:47:D6:80 (unknown)
```

6.1.3.2 使用 demo 测试验证

使用 demo 的 bt_test 进行手机连接测试。测试之前要检查 bt_init.sh 脚本是否与方案对应。路径为 tina/target/allwinner/r328s2-std/base-files/etc/bluetooth/bt_init.sh。XR829 使用的 xradio_bt_init.sh 脚本。如果不对应，需要将目录/tina/target/allwinner/r528-evb2/base-files/etc/bluetooth/bt_init.sh 的脚本换成对应方案的脚本。

使用下面的指令进入 demo 测试。

```
bt_test -i -d3       //进入蓝牙测试
```

用手机连接蓝牙，播放音乐验证 BT 是否正常。

用 LightBlue 手机 APP 连接蓝牙，通过 connect、read、write 特征值验证 BLE 是否正常。

如遇到异常问题，请开发者参考最后一章**常见问题排查指南**分析。

6.2 RTL8723DS 模组移植

主控：R329
无线模组：RTL8723DS
内核版本：linux-4.9
方案：r329-evb5_v1

6.2.1 确认硬件的工作条件配置软件输出

该部分的硬件条件需要关心的有：电源、复位、休眠唤醒、UART、主次时钟。然后对照硬件原理图，确认相应的引脚，在进行设备树配置。RTL8723DS 的原理图如下所示。

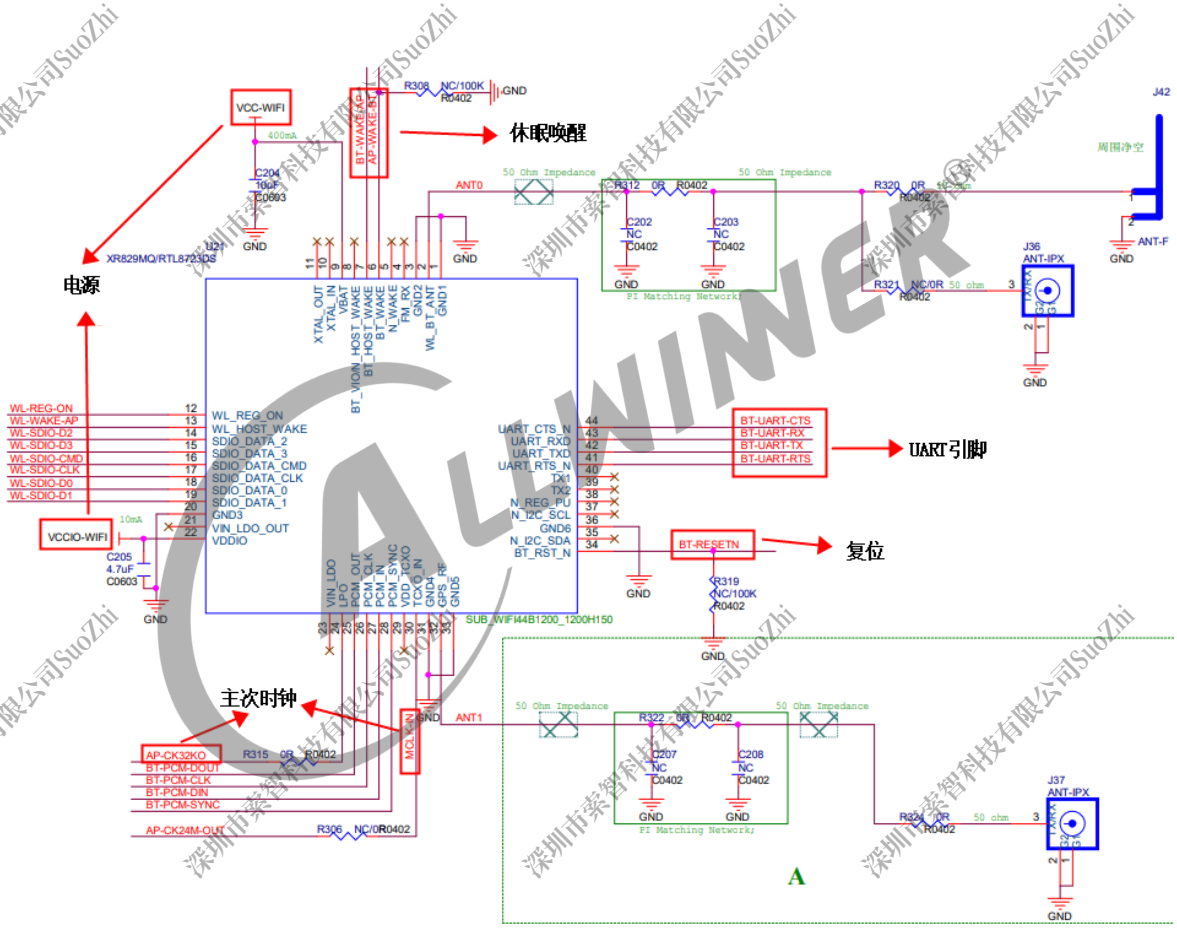

图 6-7：RTL8723DS 原理图

6.2.1.1 电源

根据 R329 电源树找到 RTL8723DS 供电 VCC-WIFI 和 VCCIO-WIFI 在 DCDC3 3.3V，属于默认开启的电源，不需要额外配置。

图 6-8: R329 电源树

6.2.1.2 复位

根据原理图中找到 BT-RESETN 连接的是 GPIO PM02，因此设备树文件中配置 bt_rst_n 标识为 PM 2, 默认输出低电平。

配置文件路径是/tina/device/config/chips/r329/configs/evb5_v1/board.dts

```
bt:bt {
    compatible = "allwinner,sunxi-bt";
    clocks = <&clk_losc_out>;
    bt_rst_n = <&r_pio PM 2 1 1 2 0>;   //复位引脚配置
    status = "okay";
};
```

6.2.1.3 休眠唤醒

根据原理图中找到 bt_wake 和 bt_hostwake 分别对应 GPIO PM01、GPIO PM03，注意配置 dts 时 bt_wake 默认输出高电平。

配置文件路径是/tina/device/config/chips/r329/configs/evb5_v1/board.dts

```
btlpm:btlpm {
    compatible = "allwinner,sunxi-btlpm";
    uart_index = <1>;
    bt_wake = <&r_pio PM 1 1 1 2 1>;
    bt_hostwake = <&r_pio PM 3 6 0 0 0>;
    status = "okay";
};
```

6.2.1.4 主次时钟

根据原理图看到主时钟由外部 24M 晶振提供，次时钟是主控输出 32k 时钟。

配置文件路径是/tina/device/config/chips/r329/configs/evb5_v1/board.dts

```
bt:bt {
    compatible = "allwinner,sunxi-bt";
    clocks = <&clk_losc_out>;      //主控输出32k时钟
    bt_rst_n = <&r_pio PM 2 1 1 2 0>;
    status = "okay";
};
```

6.2.1.5 UART 接口配置

根据原理图中找到 UART 引脚 PG06 PG07 PG08 PG09，在设备树里进行配置并使能。

```
文件tina/lichee/linux-4.9/arch/arm64/boot/dts/sunxi/sun50iw10p1-pinctrl.dtsi
uart1_pins_a: uart1@0 {
    allwinner,pins = "PG6", "PG7", "PG8", "PG9";
    allwinner,pname = "uart1_tx", "uart1_rx",
    "uart1_rts", "uart1_cts";
    allwinner,function = "uart1";
    allwinner,muxsel = <2>;
    allwinner,drive = <1>;
    allwinner,pull = <1>;
};
uart1_pins_b: uart1@1 {
    allwinner,pins = "PG6", "PG7", "PG8", "PG9";
    allwinner,function = "io_disabled";
    allwinner,muxsel = <7>;
    allwinner,drive = <1>;
    allwinner,pull = <0>;
};

文件tina/lichee/linux-4.9/arch/arm64/boot/dts/sunxi/sun50iw11p1.dtsi
uart1: uart@02500400 {
    compatible = "allwinner,sun50i-uart";
    device_type = "uart1";
    reg = <0x0 0x02500400 0x0 0x400>;
    interrupts = <GIC_SPI 3 IRQ_TYPE_LEVEL_HIGH>;
    sunxi,uart-fifosize = <256>;
    clocks = <&clk_uart1>;
    use_rxdma = <0>;
    pinctrl-names = "default", "sleep";
    pinctrl-0 = <&uart1_pins_a>;
    pinctrl-1 = <&uart1_pins_b>;
    uart1_port = <1>;
    uart1_type = <4>;
    status = "okay";
};
```

然后在内核配置里选上 UARTH5 驱动，make kernel_menuconfig->Networking support —>Bluetooth device drivers

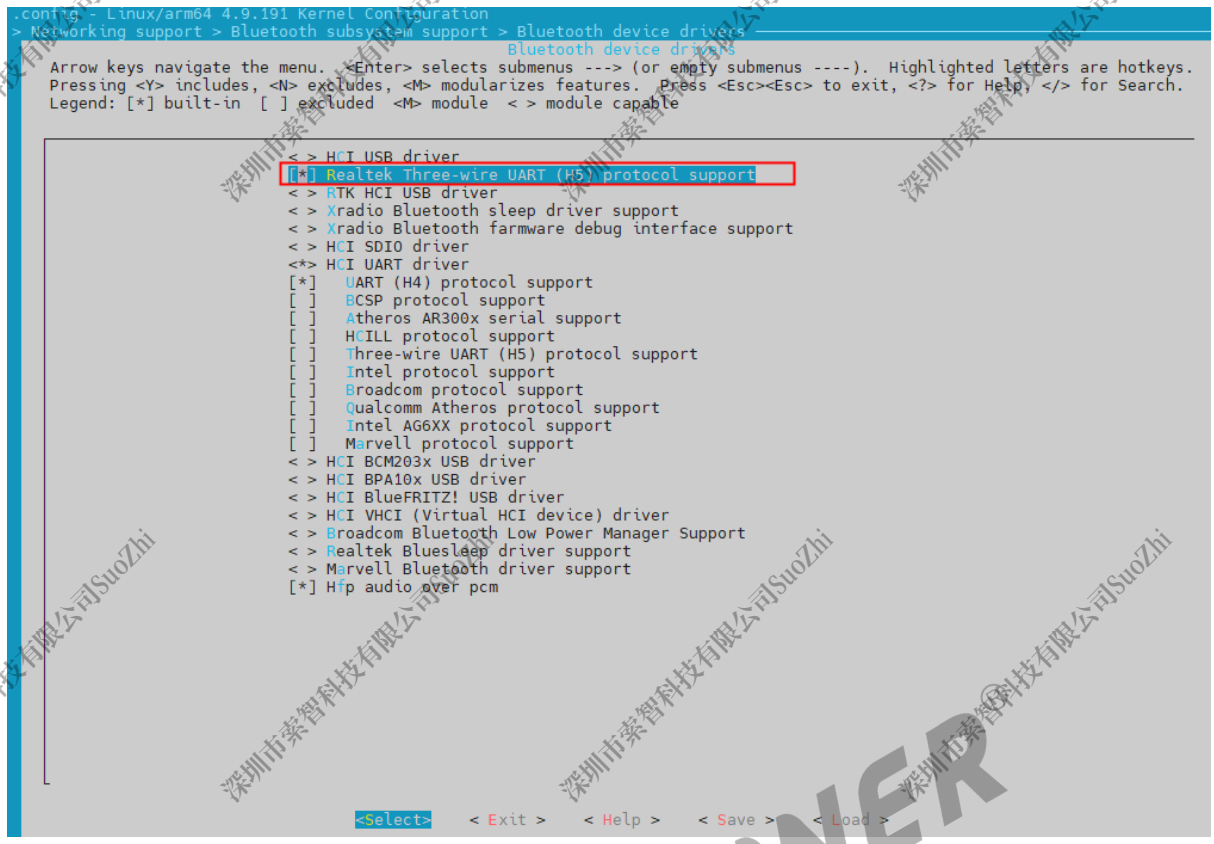

图 6-9: RTL8723DS 串口驱动

6.2.2 内核配置和用户空间配置

通过界面配置的方式打开相应的选项，分为协议栈内核部分的配置和协议栈用户空间的配置。注意有些默认配置与本方案冲突的需取消勾选。下图的配置不仅展示了蓝牙协议栈部分，也包括 R329 适配 RTL8723DS 的其他重点配置：uart 驱动、fireware、kernel 模块、rf 测试工具。

（1）协议栈内核空间的配置

```
make kernel_menuconfig
[*] Networking support  --->
  <*>   Bluetooth subsystem support  --->
    [*]   Bluetooth Classic (BR/EDR) features //经典蓝牙特性支持
    <*>     RFCOMM protocol support    //如果需要SPP文件传输等相关功能可以选上
    [*]       RFCOMM TTY support
    < >     BNEP protocol support
    < >     HIDP protocol support
    [ ]     Bluetooth High Speed (HS) features
    [*]   Bluetooth Low Energy (LE) features   //低功耗蓝牙特性支持
    [ ]   Enable LED triggers
    [ ]   Bluetooth self testing support
    [*]   Export Bluetooth internals in debugfs   //蓝牙调试节点
          Bluetooth device drivers  --->
          <*> Realtek Three-wire UART (5) protocol support   //rtk UART h5驱动
          <*> HCI UART driver
          [*]   UART (H4) protocol support    //hci uart h4驱动
```

（2）协议栈用户空间的配置

```
make menuconfig
Firmware  --->
  -*- r8723ds-firmware
Kernel modules  --->
  Wireless Drivers  --->
    <*> kmod-net-rtl8723ds
Utilities  --->
  -*- bluez-daemon
  -*- bluez-utils             //协议栈调试工具
  -*- bluez-utils-extra
  rf test tool --->
    <*> realtek-rftest        //realtek rf测试工具
  rf_hciattach --->
    <*> rf_hciattach          //hciattach
```

6.2.3 调试验证

配置完编译打包生成固件烧录到设备中，准备测试验证 RTL8723DS 蓝牙功能。

6.2.3.1 分步骤测试验证

```
1. 硬件工作条件：供电使能
echo 0 > /sys/class/rfkill/rfkill0/state
echo 1 > /sys/class/rfkill/rfkill0/state //蓝牙上电使能
2. 蓝牙初始化
rtk_hciattach -n -s 115200 /dev/ttyS1 rtk_h5 &
3. 网卡启动
hciconfig -a          //查询状态
hciconfig hci0 up
hciconfig -a
4. 启动协议栈
bluetoothd -n -d &
5. 验证功能
hcitool -i hci0 scan          //扫描周围蓝牙设备
hcitool -i hci0 lescan        //扫描周围LE设备
```

按照上面的指令在终端执行正常 log 如下：

```
root@TinaLinux:/# echo 0 > /sys/class/rfkill/rfkill0/state
[   45.176671] sunxi-bt soc@03000000:bt: block state already is 1
root@TinaLinux:/# echo 1 > /sys/class/rfkill/rfkill0/state   //蓝牙上电使能
[   52.055378] sunxi-bt soc@03000000:bt: set block: 0        //蓝牙初始化
root@TinaLinux:/# rtk_hciattach -n -s 115200 /dev/ttyS1 rtk_h5 &
root@TinaLinux:/# Realtek Bluetooth :Realtek Bluetooth init uart with init speed:115200,
    type:HCI UART H5
Realtek Bluetooth :Realtek hciattach version 3.1

Realtek Bluetooth :Use epoll
Realtek Bluetooth :[SYNC] Get SYNC Resp Pkt
Realtek Bluetooth :[CONFIG] Get SYNC pkt
Realtek Bluetooth :[CONFIG] Get CONFG pkt
Realtek Bluetooth :[CONFIG] Get CONFG resp pkt
```

```
Realtek Bluetooth :dic is 1, cfg field 0x14
Realtek Bluetooth :H5 init finished

Realtek Bluetooth :Realtek H5 IC
Realtek Bluetooth :Receive cmd complete event of command: 1001
Realtek Bluetooth :HCI Version 0x08
Realtek Bluetooth :HCI Revision 0x000d
Realtek Bluetooth :LMP Subversion 0x8723
Realtek Bluetooth :Receive cmd complete event of command: fc6d
Realtek Bluetooth :Read ROM version 02
Realtek Bluetooth :LMP Subversion 0x8723
Realtek Bluetooth :EVersion 2
Realtek Bluetooth :IC: RTL8723DS
Realtek Bluetooth :Firmware/config: rtl8723d_fw, rtl8723d_config
Realtek Bluetooth :Couldnt open extra config /opt/rtk_btconfig.txt, No such file or
    directory
Realtek Bluetooth :Couldnt access customer BT MAC file /opt/bdaddr
Realtek Bluetooth :Origin cfg len 48
Realtek Bluetooth :55 ab 23 87 2a 00 0c 00 10 02 80 92 04 50 c5 ea
Realtek Bluetooth :19 e1 1b fd af 5f 01 a4 0b d9 00 01 0f e4 00 01
Realtek Bluetooth :08 f3 00 01 0c f4 00 08 01 00 01 90 00 00 09 04
Realtek Bluetooth :Config baudrate: 04928002
Realtek Bluetooth :uart flow ctrl: 1
Realtek Bluetooth :Vendor baud from Config file: 04928002
Realtek Bluetooth :New cfg len 48
Realtek Bluetooth :55 ab 23 87 2a 00 0c 00 10 02 80 92 04 50 c5 ea
Realtek Bluetooth :19 e1 1b fd af 5f 01 a4 0b d9 00 01 0f e4 00 01
Realtek Bluetooth :08 f3 00 01 0c f4 00 08 01 00 01 90 00 00 09 04
Realtek Bluetooth :Load FW /lib/firmware/rtlbt/rtl8723d_fw OK, size 52096
Realtek Bluetooth :rtb_get_fw_project_id: opcode 0, len 1, data 9
Realtek Bluetooth :FW version 0xaa7add92, Patch num 3
Realtek Bluetooth :Chip id 0x0001
Realtek Bluetooth :Chip id 0x0002
Realtek Bluetooth :Chip id 0x0003
Realtek Bluetooth :Patch length 0x81b8
Realtek Bluetooth :Start offset 0x00004980
Realtek Bluetooth :Svn version:    20318
Realtek Bluetooth :Coexistence: BTCOEX_20180125-2323

Realtek Bluetooth :FW  exists, Config file  exists
Realtek Bluetooth :Total len 33256 for fwc
Realtek Bluetooth :baudrate in change speed command: 0x02 0x80 0x92 0x04
Realtek Bluetooth :Receive cmd complete event of command: fc17
Realtek Bluetooth :Received cc of vendor change baud
Realtek Bluetooth :Final speed 1500000
Realtek Bluetooth :end_idx: 131, lp_len: 244, additional pkts: 0

Realtek Bluetooth :Start downloading...
Realtek Bluetooth :Send last pkt
Realtek Bluetooth :Enable host hw flow control
Realtek Bluetooth :h5_hci_reset: Issue hci reset cmd
Realtek Bluetooth :Receive cmd complete event of command: 0c03
Realtek Bluetooth :Received cc of hci reset cmd
Realtek Bluetoo[   58.412657] Bluetooth: hci_uart_tty_ioctl HCIUARTSETPROTO
th :Init Process finished
[   58.421487] Bluetooth: h5_open
[   58.427194] Bluetooth: hci_uart_register_dev
Realtek Bluetooth :Realtek Bluetooth post process
Realtek Bluetooth :Device setup complete
```

```
root@TinaLinux:/# hciconfig -a       //hci查询状态
hci0:   Type: Primary  Bus: UART
        BD Address: 7C:A7:B0:26:50:65  ACL MTU: 1021:8  SCO MTU: 255:12
        DOWN
        RX bytes:1067 acl:0 sco:0 events:30 errors:0
        TX bytes:859 acl:0 sco:0 commands:30 errors:0
        Features: 0xff 0xff 0xff 0xfe 0xdb 0xfd 0x7b 0x87
        Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
        Link policy: RSWITCH HOLD SNIFF PARK
        Link mode: SLAVE ACCEPT

root@TinaLinux:/# hciconfig hci0 up     //启动hci0状态
[   93.668552] Bluetooth: hu ffffffc0049ae400 retransmitting 1 pkts
root@TinaLinux:/# hciconfig -a          //hci0查询状态
hci0:   Type: Primary  Bus: UART
        BD Address: 7C:A7:B0:26:50:65  ACL MTU: 1021:8  SCO MTU: 255:12
        UP RUNNING
        RX bytes:2133 acl:0 sco:0 events:60 errors:0
        TX bytes:1543 acl:0 sco:0 commands:61 errors:0
        Features: 0xff 0xff 0xff 0xfe 0xdb 0xfd 0x7b 0x87
        Packet type: DM1 DM3 DM5 DH1 DH3 DH5 HV1 HV2 HV3
        Link policy: RSWITCH HOLD SNIFF PARK
        Link mode: SLAVE ACCEPT
[  104.812536] Bluetooth: hu ffffffc0049ae400 retransmitting 1 pkts
        Name: 'RTK_BT_4.1'
        Class: 0x000000
        Service Classes: Unspecified
        Device Class: Miscellaneous,
        HCI Version: 4.1 (0x7)  Revision: 0xaa7a
        LMP Version: 4.1 (0x7)  Subversion: 0xdd92
        Manufacturer: Realtek Semiconductor Corporation (93)

root@TinaLinux:/# bluetoothd -n -d &     //启动蓝牙协议栈
root@TinaLinux:/# bluetoothd[1545]: Bluetooth daemon 5.54
bluetoothd[1545]: src/main.c:parse_config() parsing /etc/bluetooth/main.conf
bluetoothd[1545]: src/main.c:parse_config() Key file does not have key 'DiscoverableTimeout
    ' in group 'General'
bluetoothd[1545]: src/main.c:parse_config() Key file does not have key 'AlwaysPairable' in
    group 'General'
bluetoothd[1545]: src/main.c:parse_config() Key file does not have key 'PairableTimeout' in
     group 'General'
bluetoothd[1545]: src/main.c:parse_config() Key file does not have key 'Privacy' in group '
    General'
bluetoothd[1545]: src/main.c:parse_config() Key file does not have key 'JustWorksRepairing'
     in group 'General'
bluetoothd[1545]: src/main.c:parse_config() Key file does not have key 'Name' in group '
    General'
bluetoothd[1545]: src/main.c:parse_config() Key file does not have key 'Class' in group '
    General'
bluetoothd[1545]: src/main.c:parse_config() Key file does not have key 'DeviceID' in group
    'General'
bluetoothd[1545]: src/main.c:parse_config() Key file does not have key '
    ReverseServiceDiscovery' in group 'General'
bluetoothd[1545]: src/main.c:parse_config() Key file does not have key 'Cache' in group '
    GATT'
bluetoothd[1545]: src/main.c:parse_config() Key file does not have key 'KeySize' in group '
    GATT'
bluetoothd[1545]: src/main.c:parse_config() Key file does not have key 'ExchangeMTU' in
    group 'GATT'
bluetoothd[1545]: src/main.c:parse_config() Key file does not have key 'Channels' in group
```

```
    'GATT'
bluetoothd[1545]: src/adapter.c:adapter_init() sending read version command
bluetoothd[1545]: Starting SDP server
bluetoothd[1545]: src/sdpd-service.c:register_device_id() Adding device id record for
    0002:1d6b:0246:0536
bluetoothd[1545]: src/plugin.c:plugin_init() Loading builtin plugins
bluetoothd[1545]: src/plugin.c:add_plugin() Loading hostname plugin
bluetoothd[1545]: src/plugin.c:add_plugin() Loading wiimote plugin
bluetoothd[1545]: src/plugin.c:add_plugin() Loading autopair plugin
bluetoothd[1545]: src/plugin.c:add_plugin() Loading policy plugin
bluetoothd[1545]: src/plugin.c:add_plugin() Loading a2dp plugin
bluetoothd[1545]: src/plugin.c:add_plugin() Loading avrcp plugin
bluetoothd[1545]: src/plugin.c:add_plugin() Loading network plugin
bluetoothd[1545]: src/plugin.c:add_plugin() Loading input plugin
bluetoothd[1545]: src/plugin.c:add_plugin() Loading hog plugin
bluetoothd[1545]: src/plugin.c:add_plugin() Loading gap plugin
bluetoothd[1545]: src/plugin.c:add_plugin() Loading scanparam plugin
bluetoothd[1545]: src/plugin.c:add_plugin() Loading deviceinfo plugin
bluetoothd[1545]: src/plugin.c:add_plugin() Loading battery plugin
bluetoothd[1545]: src/plugin.c:plugin_init() Loading plugins /usr/lib/bluetooth/plugins
bluetoothd[1545]: Failed to init battery plugin
bluetoothd[1545]: profiles/input/suspend-none.c:suspend_init()
bluetoothd[1545]: profiles/network/manager.c:read_config() /etc/bluetooth/network.conf: Key
     file does not have key 'DisableSecurity' in group 'General'
bluetoothd[1545]: profiles/network/manager.c:read_config() Config options: Security=true
bluetoothd[1545]: kernel lacks bnep-protocol support
bluetoothd[1545]: System does not support network plugin
bluetoothd[1545]: src/main.c:main() Entering main loop
bluetoothd[1545]: src/rfkill.c:rfkill_event() RFKILL event idx 0 type 2 op 0 soft 0 hard 0
bluetoothd[1545]: Bluetooth management interface 1.14 initialized
bluetoothd[1545]: src/adapter.c:read_version_complete() sending read supported commands
    command
bluetoothd[1545]: src/adapter.c:read_version_complete() sending read index list command
bluetoothd[1545]: src/rfkill.c:rfkill_event() RFKILL event idx 1 type 1 op 0 soft 0 hard 0
bluetoothd[1545]: src/rfkill.c:rfkill_event() RFKILL event idx 2 type 2 op 0 soft 0 hard 0
bluetoothd[1545]: src/adapter.c:read_commands_complete() Number of commands: 65
bluetoothd[1545]: src/adapter.c:read_commands_complete() Number of events: 35
bluetoothd[1545]: src/adapter.c:read_commands_complete() enabling kernel-side connection
    control
bluetoothd[1545]: src/adapter.c:read_index_list_complete() Number of controllers: 1
bluetoothd[1545]: src/adapter.c:read_index_list_complete() Found index 0
bluetoothd[1545]: src/adapter.c:index_added() index 0
bluetoothd[1545]: src/adapter.c:btd_adapter_new() System name: BlueZ 5.54
bluetoothd[1545]: src/adapter.c:btd_adapter_new() Major class: 0
bluetoothd[1545]: src/adapter.c:btd_adapter_new() Minor class: 0
bluetoothd[1545]: src/adapter.c:btd_adapter_new() Modalias: usb:v1D6Bp0246d0536
bluetoothd[1545]: src/adapter.c:btd_adapter_new() Discoverable timeout: 180 seconds
bluetoothd[1545]: src/adapter.c:btd_adapter_new() Pairable timeout: 0 seconds
bluetoothd[1545]: src/adapter.c:index_added() sending read info command for index 0
bluetoothd[1545]: src/adapter.c:read_info_complete() index 0 status 0x00
bluetoothd[1545]: src/adapter.c:clear_uuids() sending clear uuids command for index 0
bluetoothd[1545]: src/adapter.c:clear_devices() sending clear devices command for index 0
bluetoothd[1545]: src/adapter.c:set_mode() sending set mode command for index 0
bluetoothd[1545]: src/adapter.c:set_mode() sending set mode command for index 0
bluetoothd[1545]: src/adapter.c:set_mode() sending set mode command for index 0
bluetoothd[1545]: src/adapter.c:set_privacy() sending set privacy command for index 0
bluetoothd[1545]: src/adapter.c:set_privacy() setting privacy mode 0x00 for index 0
bluetoothd[1545]: src/advertising.c:btd_adv_manager_new() LE Advertising Manager created
    for adapter: /org/bluez/hci0
bluetoothd[1545]: profiles/audio/a2dp.c:media_server_probe() path /org/bluez/hci0
```

```
bluetoothd[1545]: plugins/hostname.c:hostname_probe()
bluetoothd[1545]: profiles/audio/avrcp.c:avrcp_controller_server_probe() path /org/bluez/
    hci0
bluetoothd[1545]: src/adapter.c:adapter_service_add() /org/bluez/hci0
bluetoothd[1545]: src/sdpd-service.c:add_record_to_server() Adding record with handle 0
    x10001
bluetoothd[1545]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00000017-0000-1000-8000-00805f9
bluetoothd[1545]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00000100-0000-1000-8000-00805f9
bluetoothd[1545]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00001002-0000-1000-8000-00805f9
bluetoothd[1545]: src/sdpd-service.c:add_record_to_server() Record pattern UUID 0000110e
    -0000-1000-8000-00805f9
bluetoothd[1545]: src/sdpd-service.c:add_record_to_server() Record pattern UUID 0000110f
    -0000-1000-8000-00805f9
bluetoothd[1545]: src/adapter.c:adapter_service_insert() /org/bluez/hci0
bluetoothd[1545]: src/adapter.c:add_uuid() sending add uuid command for index 0
bluetoothd[1545]: profiles/audio/avrcp.c:avrcp_target_server_probe() path /org/bluez/hci0
bluetoothd[1545]: src/adapter.c:adapter_service_add() /org/bluez/hci0
bluetoothd[1545]: src/sdpd-service.c:add_record_to_server() Adding record with handle 0
    x10002
bluetoothd[1545]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00000017-0000-1000-8000-00805f9
bluetoothd[1545]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00000100-0000-1000-8000-00805f9
bluetoothd[1545]: src/sdpd-service.c:add_record_to_server() Record pattern UUID
    00001002-0000-1000-8000-00805f9
bluetoothd[1545]: src/sdpd-service.c:add_record_to_server() Record pattern UUID 0000110c
    -0000-1000-8000-00805f9
bluetoothd[1545]: src/sdpd-service.c:add_record_to_server() Record pattern UUID 0000110e
    -0000-1000-8000-00805f9
bluetoothd[1545]: src/adapter.c:adapter_service_insert() /org/bluez/hci0
bluetoothd[1545]: src/adapter.c:add_uuid() sending add uuid command for index 0
bluetoothd[1545]: profiles/audio/a2dp.c:a2dp_sink_server_probe() path /org/bluez/hci0
bluetoothd[1545]: profiles/audio/a2dp.c:a2dp_source_server_probe() path /org/bluez/hci0
bluetoothd[1545]: src/adapter.c:btd_adapter_unblock_address() hci0 00:00:00:00:00:00
bluetoothd[1545]: src/adapter.c:get_ltk_info() 40:26:19:94:25:B0
bluetoothd[1545]: src/adapter.c:get_slave_ltk_info() 40:26:19:94:25:B0
bluetoothd[1545]: src/device.c:device_create_from_storage() address 40:26:19:94:25:B0
bluetoothd[1545]: src/device.c:device_new() address 40:26:19:94:25:B0
bluetoothd[1545]: src/device.c:device_new() Creating device /org/bluez/hci0/
    dev_40_26_19_94_25_B0
bluetoothd[1545]: src/device.c:btd_device_set_temporary() temporary 0
bluetoothd[1545]: src/device.c:device_set_bonded()
bluetoothd[1545]: src/adapter.c:load_link_keys() hci0 keys 1 debug_keys 0
bluetoothd[1545]: src/adapter.c:load_ltks() hci0 keys 0
bluetoothd[1545]: src/adapter.c:load_irks() hci0 irks 0
bluetoothd[1545]: src/adapter.c:load_conn_params() hci0 conn params 0
bluetoothd[1545]: src/device.c:device_probe_profiles() Probing profiles for device
    40:26:19:94:25:B0
bluetoothd[1545]: profiles/audio/avrcp.c:avrcp_controller_probe() path /org/bluez/hci0/
    dev_40_26_19_94_25_B0
bluetoothd[1545]: profiles/audio/control.c:control_init() Registered interface org.bluez.
    MediaControl1 on path /org/bluez/hci0/dev_40_26_19_94_25_B0
bluetoothd[1545]: src/service.c:btd_service_ref() 0x2fdabf00: ref=2
bluetoothd[1545]: src/service.c:change_state() 0x2fdabf00: device 40:26:19:94:25:B0 profile
     avrcp-controller state changed: unavailable -> disconnected (0)
bluetoothd[1545]: profiles/audio/avrcp.c:avrcp_target_probe() path /org/bluez/hci0/
    dev_40_26_19_94_25_B0
```

```
bluetoothd[1545]: src/service.c:btd_service_ref() 0x2fd9df50: ref=2
bluetoothd[1545]: src/service.c:change_state() 0x2fd9df50: device 40:26:19:94:25:B0 profile
     audio-avrcp-target state changed: unavailable -> disconnected (0)
bluetoothd[1545]: profiles/audio/a2dp.c:a2dp_source_probe() path /org/bluez/hci0/
    dev_40_26_19_94_25_B0
bluetoothd[1545]: profiles/audio/source.c:source_init() /org/bluez/hci0/
    dev_40_26_19_94_25_B0
bluetoothd[1545]: src/service.c:btd_service_ref() 0x2fdacf70: ref=2
bluetoothd[1545]: src/service.c:change_state() 0x2fdacf70: device 40:26:19:94:25:B0 profile
     a2dp-source state changed: unavailable -> disconnected (0)
bluetoothd[1545]: src/adapter.c:load_connections() sending get connections command for
    index 0
bluetoothd[1545]: src/adapter.c:adapter_service_insert() /org/bluez/hci0
bluetoothd[1545]: src/adapter.c:add_uuid() sending add uuid command for index 0
bluetoothd[1545]: src/adapter.c:set_did() hci0 source 2 vendor 1d6b product 246 version 536
bluetoothd[1545]: src/adapter.c:adapter_register() Adapter /org/bluez/hci0 registered
bluetoothd[1545]: src/adapter.c:set_dev_class() sending set device class command for index
    0
bluetoothd[1545]: src/adapter.c:set_name() sending set local name command for index 0
bluetoothd[1545]: src/adapter.c:adapter_start() adapter /org/bluez/hci0 has been enabled
[  160.984530] Bluetooth: hu ffffffc0049ae400 retransmitting 1 pkts
bluetoothd[1545]: src/adapter.c:new_settings_callback() Settings: 0x000000c1
bluetoothd[1545]: src/adapter.c:settings_changed() Changed settings: 0x00000040
bluetoothd[1545]: src/adapter.c:settings_changed() Pending settings: 0x00000000
bluetoothd[1545]: src/adapter.c:new_settings_callback() Settings: 0x000002c1
bluetoothd[1545]: src/adapter.c:settings_changed() Changed settings: 0x00000200
bluetoothd[1545]: src/adapter.c:settings_changed() Pending settings: 0x00000000
bluetoothd[1545]: src/adapter.c:trigger_passive_scanning()
bluetoothd[1545]: src/adapter.c:new_settings_callback() Settings: 0x00000ac1
bluetoothd[1545]: src/adapter.c:settings_changed() Changed settings: 0x00000800
bluetoothd[1545]: src/adapter.c:settings_changed() Pending settings: 0x00000000
bluetoothd[1545]: Failed to set privacy: Rejected (0x0b)
bluetoothd[1545]: src/adapter.c:add_whitelist_complete() 40:26:19:94:25:B0 added to kernel
    whitelist
bluetoothd[1545]: src/adapter.c:load_link_keys_complete() link keys loaded for hci0
bluetoothd[1545]: src/adapter.c:load_ltks_complete() LTKs loaded for hci0
bluetoothd[1545]: src/adapter.c:load_irks_complete() IRKs loaded for hci0
bluetoothd[1545]: src/adapter.c:load_conn_params_complete() Connection Parameters loaded
    for hci0
bluetoothd[1545]: src/adapter.c:get_connections_complete() Connection count: 0
bluetoothd[1545]: src/adapter.c:local_name_changed_callback() Name: aw-bt-test-50-65
bluetoothd[1545]: src/adapter.c:local_name_changed_callback() Short name:
bluetoothd[1545]: src/adapter.c:local_name_changed_callback() Current alias: aw-bt-test
    -50-65
root@TinaLinux:/# hcitool -i hci0 scan      //扫描周围蓝牙设备
Scanning ...
[  349.828552] Bluetooth: hu ffffffc0049ae400 retransmitting 1 pkts
        F0:62:5A:CE:60:1E       n/a
        E0:DC:FF:E9:34:1E       flyBT
        D8:6C:02:CA:DB:9E       meizu X8
        58:85:E9:32:20:24       n/a.u X8
root@TinaLinux:/# hcitool -i hci0 lescan       //扫描周围LE设备
[  430.964495] Bluetooth: hu ffffffc0049ae400 retransmitting 1 pkts
LE Scan ...
6D:35:06:67:5F:E8 (unknown)
60:C3:35:A1:72:64 (unknown)
FC:E8:00:10:58:8E EDIFIER BLE
60:46:00:54:48:54 (unknown)
```

6.2.3.2 使用 demo 测试验证

使用 demo 测试之前要检查 bt_init.sh 脚本，使用 RTK 方案的脚本。路径为 tina/target/allwinner/r329-evb5_v1/base-files/etc/bluetooth/bt_init.sh。

进入 tina 系统，使用 bt_test 进行测试。

```
adb shell //进入tina系统
bt_test -i -d3//进入蓝牙测试
```

用手机连接蓝牙，播放音乐验证 BT 是否正常。

用 LightBlue 手机 APP 连接 LE，通过 connect、read、write 特征值验证 BLE 是否正常。

如遇到异常问题，请开发者参考最后一章**常见问题排查指南**分析。

7 常见问题排查指南

7.1 排查思路

如果 hci0 无法启动或者异常，可以按照以下顺序依次进行确认排查。

序号	说明	检查思路
1	检查两路电源供电,VCC/VCCIO	万用表测量
2	检查 BT-RESETN 是否拉高，符合时序要求	万用表测量
3	检查 BT-WAKE-BT 是否拉高（可选）	万用表测量
4	检查 UART 功能是否正常	单独测试自发自收，保证 uart 节点通信正常
5	检查主时钟晶振是否正常	示波器测量
6	检查次时钟晶振是否正常（可选）	示波器测量
7	检查固件下载是否正常	检查文件路径，启动时序，uart 配置（波特率，流控等）

7.2 常见问题示例

7.2.1 播放音乐失败

问题现象：手机可以连接成功设备蓝牙，但是播放音乐失败，出现如下异常 log。（出现方案：R328+XR829）

图 7-1: 异常 log

问题原因：设备端没有设置 default 声卡，导致 PCM 打开失败。声卡配置一般由相应的技术负责人配置，配置后设备端有配置文件 /etc/asound.conf。

解决方案：可以检查下设备端是否有配置文件 /etc/asound.conf，如果没有则选上下图的配置 <*>alsa-conf-aw

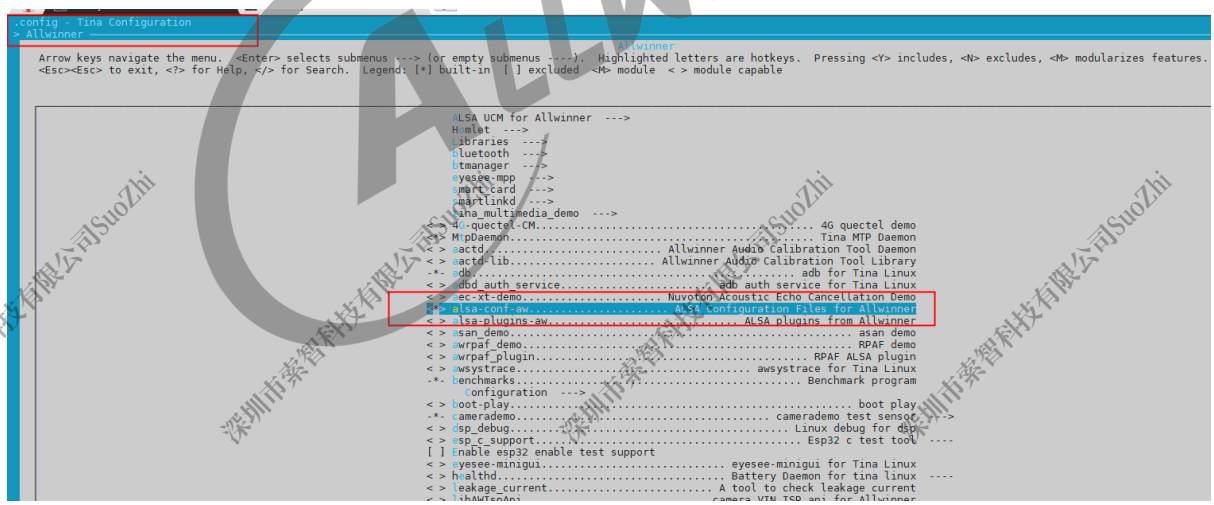

图 7-2: 解决方案

7.2.2 蓝牙初始化超时

7.2.2.1 R528+XR819s

执行 hciattach -n ttyS1 xr819s & 失败，显示 Initialization timed out. 如图所示（出现方案：R528+XR819s）

```
remain len:     0x8A50.
remain len:     0x8650.
remain len:     0x8250.
remain len:     0x7E50.
remain len:     0x7A50.
remain len:     0x7650.
remain len:     0x7250.
remain len:     0x6E50.
remain len:     0x6A50.
remain len:     0x6650.
remain len:     0x6250.
remain len:     0x5E50.
remain len:     0x5A50.
remain len:     0x5650.
remain len:     0x5250.
remain len:     0x4E50.
remain len:     0x4A50.
remain len:     0x4650.
remain len:     0x4250.
remain len:     0x3E50.
remain len:     0x3A50.
remain len:     0x3650.
remain len:     0x3250.
remain len:     0x2E50.
remain len:     0x2A50.
remain len:     0x2650.
remain len:     0x2250.
remain len:     0x1E50.
remain len:     0x1A50.
remain len:     0x1650.
remain len:     0x1250.
remain len:     0x0E50.
remain len:     0x0A50.
remain len:     0x0650.
remain len:     0x0250.
remain len:     0x0000.
load firmware done.
jump:
set pc 0, val 0
Now the system will jump to 00000000
Set HW FlowControl On
userial_vendor_set_hw_fctrl set hw flowcontrol on
[xradio_init] send reset cmd...
writing
01 03 0c 00
Initialization timed out.
```

图 7-3: 蓝牙初始化失败 log

问题原因：未添加补丁 011-add-xr819s-hciattach-support.patch，导致蓝牙初始化用了 XR829 的 hciattach。

解决方案：打上补丁 011-add-xr819s-hciattach-support.patch，重新编译。

7.2.2.2 D1+AW869B

执行 hciattach -s 1500000 /dev/ttyS1 aic & 失败，显示 Initialization timed out. 如图所示（出现方案：D1+AW869B）

```
root@TinaLinux:/# ./tmp/hciattach -s 1500000 /dev/ttyS1 aic
AIC Bluetooth init uart with init speed:1500000, final_speed:1500000, type:HCI UART H4
AIC Bluetooth: aic_send_hci_cmd
AIC Bluetooth: 03 0c 00
Initialization timed out.
root@TinaLinux:/#
```

图 7-4: AW869 蓝牙初始化失败

问题原因： 一种原因可能是初始波特率不是 1500000 导致通信不上，另一种原因可能是 AW869B 里面的固件不对或者没有。

解决方案：第一步：确定初始波特率是不是 1500000；第二步：打开 m menuconfig -> firmware 固件是否选对，而且要取消别的方案的固件；

第三步：看上电启动的 log，里面找到加载蓝牙固件的路径是否正确，正常路径是/lib/fireware/xxx，如图所示。

```
[   11.570073] mmc1: new SDIO card at address 1265
[   11.609862] aicbsp: aicbsp_sdio_probe:1
[   11.619866] aicbsp: aicbsp_sdio_probe:2
[   11.624165] aicbsp: aicbsp_sdio_probe after replace:1
[   11.639980] sunxi-mmc 4021000.sdmmc: sdc set ios:clk 70000000Hz bm PP pm ON vdd 21 width 4 timing LEGACY(SDR12) dt B
[   11.660132] sunxi-mmc 4021000.sdmmc: failed to get DS26_SDR12 used default
[   11.667808] aicbsp: Set SDIO Clock 66 MHz
[   11.690924] aicbsp: sdio_err:<aicwf_sdio_hal_irqhandler,1071>: Interrupt but no data
[   11.699686] aicbsp: aicbsp_driver_fw_init, chip rev: 3
[   11.712339] rwnx_load_firmware :firmware path = /lib/firmware/fw_adid.bin
[   11.903518] rwnx_load_firmware :firmware path = /lib/firmware/fw_patch.bin
[   12.007833] rwnx_load_firmware :firmware path = /lib/firmware/fw_patch_table.bin
[   12.054789] aicbsp: bt patch version: - Jul 30 2021 17:16:34 - git rom_release-6a86f80-dirty
[   12.076283] rwnx_load_firmware :firmware path = /lib/firmware/fmacfw.bin
[   12.375277] aicbsp: aicbsp_sdio_remove
[   12.379459] aicbsp: aicwf_sdio_release
[   12.386823] aicbsp: aicwf_bus_deinit
[   12.386830] aicbsp: aicwf_sdio_bus_stop
[   12.417758] aicbsp: sdio_err:<aicwf_sdio_bustx_thread,973>: sdio bustx thread stop
[   12.438760] aicbsp: sdio_err:<aicwf_sdio_busrx_thread,992>: sdio busrx thread stop
[   12.460088] aicbsp: aicbsp_sdio_remove done
[   12.470160] aicsdio: aicwf_sdio_probe:1
```

图 7-5: AW869B 固件下载

7.2.3 未发现 hci0

问题现象：执行完初始化后，hciconfig -a 没有发现 hci0。（出现方案：D1+AW869B）

```
root@TinaLinux:/# bluetoothd -n -d &
root@TinaLinux:/# bluetoothd[668]: Bluetooth daemon 5.54
bluetoothd[668]: src/main.c:parse_config() parsing /etc/bluetooth/main.conf
bluetoothd[668]: src/main.c:parse_config() Key file does not have key 'DiscoverableTimeout' in group 'General'
bluetoothd[668]: src/main.c:parse_config() Key file does not have key 'AlwaysPairable' in group 'General'
bluetoothd[668]: src/main.c:parse_config() Key file does not have key 'PairableTimeout' in group 'General'
bluetoothd[668]: src/main.c:parse_config() Key file does not have key 'Privacy' in group 'General'
bluetoothd[668]: src/main.c:parse_config() Key file does not have key 'JustWorksRepairing' in group 'General'
bluetoothd[668]: src/main.c:parse_config() Key file does not have key 'Name' in group 'General'
bluetoothd[668]: src/main.c:parse_config() Key file does not have key 'Class' in group 'General'
bluetoothd[668]: src/main.c:parse_config() Key file does not have key 'DeviceID' in group 'General'
bluetoothd[668]: src/main.c:parse_config() Key file does not have key 'ReverseServiceDiscovery' in group 'General'
bluetoothd[668]: src/main.c:parse_config() Key file does not have key 'Cache' in group 'GATT'
bluetoothd[668]: src/main.c:parse_config() Key file does not have key 'KeySize' in group 'GATT'
bluetoothd[668]: src/main.c:parse_config() Key file does not have key 'ExchangeMTU' in group 'GATT'
bluetoothd[668]: src/main.c:parse_config() Key file does not have key 'Channels' in group 'GATT'
bluetoothd[668]: src/adapter.c:adapter_init() sending read version command
bluetoothd[668]: Starting SDP server
bluetoothd[668]: src/sdpd-service.c:register_device_id() Adding device id record for 0002:1d6b:0246:0536
bluetoothd[668]: src/plugin.c:plugin_init() Loading builtin plugins
bluetoothd[668]: src/plugin.c:add_plugin() Loading hostname plugin
bluetoothd[668]: src/plugin.c:add_plugin() Loading wiimote plugin
bluetoothd[668]: src/plugin.c:add_plugin() Loading autopair plugin
bluetoothd[668]: src/plugin.c:add_plugin() Loading policy plugin
bluetoothd[668]: src/plugin.c:add_plugin() Loading a2dp plugin
bluetoothd[668]: src/plugin.c:add_plugin() Loading avrcp plugin
bluetoothd[668]: src/plugin.c:add_plugin() Loading network plugin
bluetoothd[668]: src/plugin.c:add_plugin() Loading input plugin
bluetoothd[668]: src/plugin.c:add_plugin() Loading hog plugin
bluetoothd[668]: src/plugin.c:add_plugin() Loading gap plugin
bluetoothd[668]: src/plugin.c:add_plugin() Loading scanparam plugin
bluetoothd[668]: src/plugin.c:add_plugin() Loading deviceinfo plugin
bluetoothd[668]: src/plugin.c:add_plugin() Loading battery plugin
bluetoothd[668]: src/plugin.c:plugin_init() Loading plugins /usr/lib/bluetooth/plugins
bluetoothd[668]: Failed to init battery plugin
bluetoothd[668]: profiles/input/suspend-none.c:suspend_init()
bluetoothd[663]: profiles/network/manager.c:read_config() /etc/bluetooth/network.conf: Key file does not have key 'DisableSecurity' in group 'General'
bluetoothd[668]: profiles/network/manager.c:read_config() Config options: Security=true
bluetoothd[658]: kernel lacks bnep-protocol support
bluetoothd[668]: System does not support network plugin
bluetoothd[668]: src/main.c:main() Entering main loop
bluetoothd[668]: src/rfkill.c:rfkill_event() RFKILL event idx 0 type 2 op 0 soft 0 hard 0
bluetoothd[668]: Bluetooth management interface 1.14 initialized
bluetoothd[668]: src/adapter.c:read_version_complete() sending read supported commands command
bluetoothd[668]: src/adapter.c:read_version_complete() sending read index list command
bluetoothd[668]: src/rfkill.c:rfkill_event() RFKILL event idx 1 type 2 op 0 soft 1 hard 0
bluetoothd[668]: src/rfkill.c:rfkill_event() RFKILL event idx 2 type 1 op 0 soft 0 hard 0
bluetoothd[668]: src/adapter.c:read_commands_complete() Number of commands: 65
bluetoothd[668]: src/adapter.c:read_commands_complete() Number of events: 35
bluetoothd[668]: src/adapter.c:read_commands_complete() enabling kernel-side connection control
bluetoothd[668]: src/adapter.c:read_index_list_complete() Number of controllers: 0

root@TinaLinux:/# hciconfig hci0 up
Can't get device info: No such device
root@TinaLinux:/# hciconfig -a
root@TinaLinux:/#
```

图 7-6: 问题现象

问题原因：蓝牙初始化不完整导致，没有使能 hci uart proc，前期 hciattach.c 有一端代码被人屏蔽导致。

```
#if 1
    tcflush(fd, TCIOFLUSH);
    /* Set actual baudrate */
    if (set_speed(fd, &ti, u->speed) < 0) {
        perror("Can't set baud rate");
        goto fail;
    }

    /* Set TTY to N_HCI line discipline */
    i = N_HCI;
    if (ioctl(fd, TIOCSETD, &i) < 0) {
        perror("Can't set line discipline");
        goto fail;
    }

    if (flags && ioctl(fd, HCIUARTSETFLAGS, flags) < 0) {
        perror("Can't set UART flags");
        goto fail;
    }

    if (ioctl(fd, HCIUARTSETPROTO, u->proto) < 0) {
        perror("Can't set device");
        goto fail;
    }

    if (u->post && u->post(fd, u, &ti) < 0)
        goto fail;
#endif
    return fd;
```

图 7-7: 使能 hci0

解决办法：检查代码是否有 Set TTY to N_HCI line discipline 这个过程，这个过程就是使能 hci uart proc，产生 hci0。如上图所示。

著作权声明

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


、 

、 
（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。