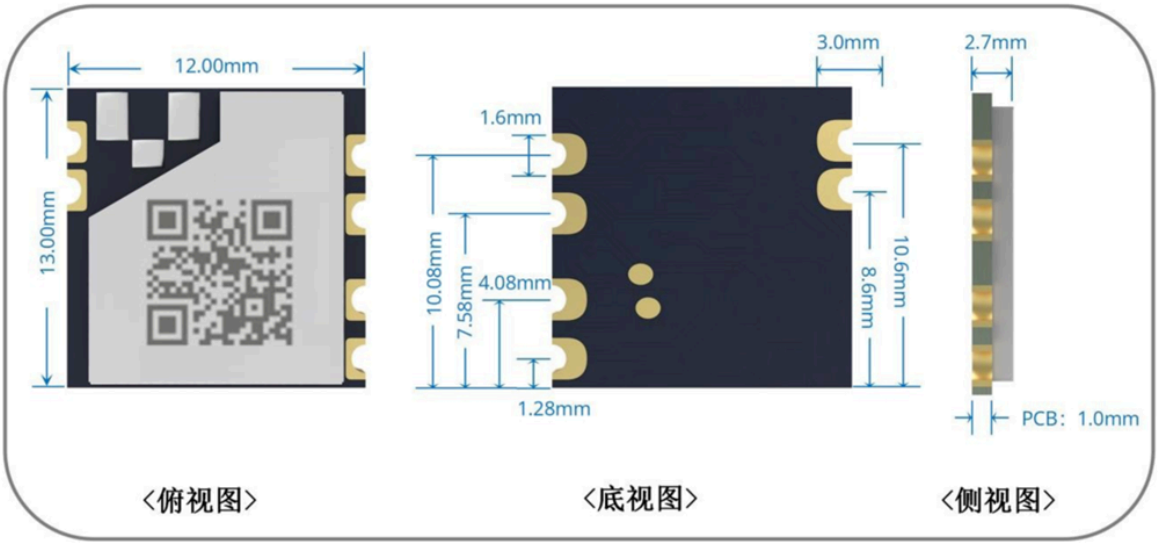
【星闪】星闪三模网卡M573H-USBDG让香橙派5 Plus秒变星闪神器！手把手教你~

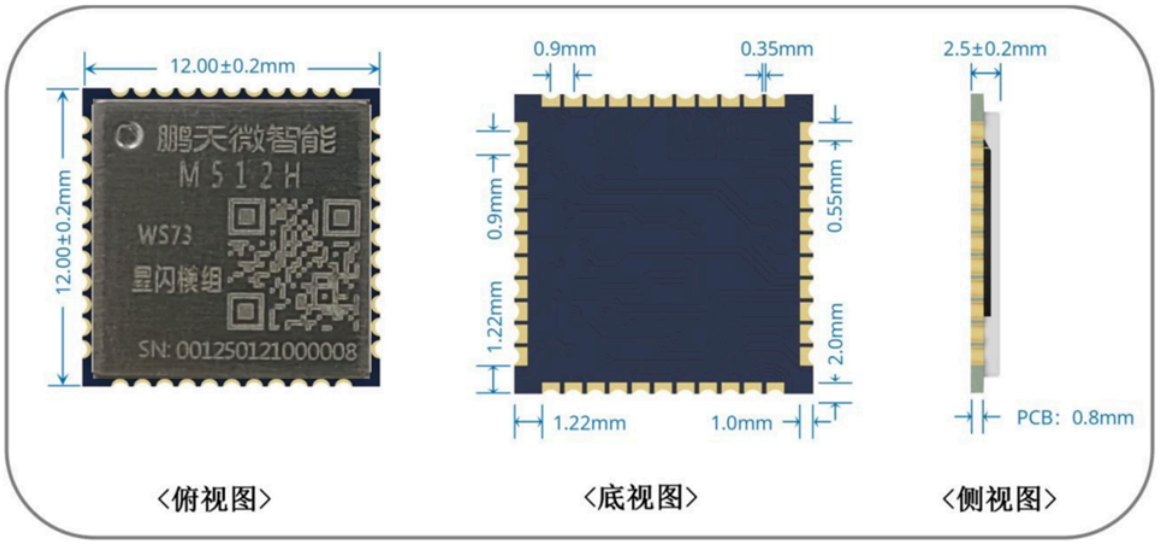
本文基于香橙派 5 Plus 与 USB 接口的 WS73 网卡模块 M573H-USBDG，手把手教你从编译到星闪通信。

简介

星闪三模网卡模块 M512H-WS73 和 **M573H-USBDG** 是深圳如般微电子有限公司（曾用名：深圳鹏天微智能科技有限公司）推出的高度集成 2.4GHz Wi-Fi、 BLE 和 SLE（星闪）的模块。M512H-WS73 和 M573H-USBDG 专为网卡应用设计，其中 M573H-USBDG 使用 USB 接口与主机通信，而 M512H-WS73 使用 SDIO 与主机通信。



M573H-USBDG模块封装尺寸



M512H-WS73模组封装尺寸图

特征

- 共同特征
  - 支持 BLE 1MHz/2MHz 频宽，支持 BLE 4.0/4.1/4.2/5.0/5.1/5.2 协议， 支持 BLE Mesh 和 BLE 网关功能，最大空口速率 2Mbps。
  - 支持 SLE 1MHz/2MHz 频宽，支持 SLE1.0 协议， 支持 SLE 网关功能，最大空口速率 4Mbps。
  - 内部芯片集成高性能 32bit 微处理器和安全处理引擎。
- M573H-USBDG
  - 支持 USB2.0 接口， 最高速率 480Mbps； 模块作为从机，通过 USB 接口搭载到主机运行。
- M573H-USBDG
  - 提供 UART 和 GPIO 接口，同时支持高速 SDIO2.0 接口，最高时钟可达 50MHz；芯片作为从机，通过 SDIO 接口搭载到主机运行。

准备工作

硬件准备

1. **PC 主机**:
   1. 用于交叉编译 RK3588 的 Linux 内核、交叉编译 WS73 驱动。
   2. 需要安装 Ubuntu 22.04（不支持 20.04）的实体机或虚拟机。

3. 不要在 WSL 内编译，《OrangePi_5_Plus_RK3588_用户手册_v2.1.pdf》内说明“没有在 WSL 虚拟机中测试过”。

2. **Orange Pi 5 Plus**：
    1. 用于驱动星闪三模网卡模块 WS73。目前 WS73 还没有 Windows 驱动。
    2. 香橙派 5 Plus 基于 RK3588 的单板计算机，4~16GB RAM 版本的均可。
    3. 推荐准备两个香橙派 5 Plus，以方便使用两个 WS73 对测（目前不支持一个单板计算机上插两个 WS73）。
    4. 其他配件：
        1. **电源适配器**，Orange Pi 5 Plus 建议使用 5V/4A 的 Type-C 电源供电。
        2. **USB 转串口模块**：用于控制香橙派 5 Plus（波特率默认 1500000）。
        3. 16GB 容量（推荐 32GB 或以上）的 class10 级或以上的 **SD 卡及读卡器**：用于烧录 RK3588 系统镜像。
        4. 两根**网线**，分别将 PC 主机和香橙派 5 Plus 通过网线插入路由器：用于高速传输数据。
3. **星闪三模网卡模块 WS73**：
    1. M512H-WS73 或 M573H-USBDG。
    2. 本文以 USB 接口的 **M573H-USBDG** 为例演示。SDIO 接口的 WS73 在软件上用起来一样，只是配置和接口不同。
    3. 推荐准备两个，以方便使用两个 WS73 对测。
4. **（可选）星闪开发板 E528H-WS63**：
    1. 如果没有两个 WS73 模块，也可以考虑使用 WS73+WS63 对测。
    2. 星闪开发板 **E528H-WS63** 是深圳如般微电子有限公司推出的基于 Hi3863 的开发板，同样支持 2.4GHz Wi-Fi、 BLE 和 SLE（星闪）。WS73 专为网卡应用而设计，不作主控，而 WS63 则可以作为 MCU 主控。
    3. 如果使用 WS63，需要多准备一个 **USB 转串口模块**，因为 AT 固件默认使用串口 1。
    4. 注意！目前 `WS73_1.10.110` SDK 更新了星闪广播包格式，而 WS63 的 `WS63_1.10.101` 或 `WS63_1.10.102` SDK 暂无更新，因此 WS73 开广播时 WS63 无法扫描和连接，只能用 WS63 做服务端开广播，WS73 做客户端连接。

软件准备

1. **PC 主机**：

1. VMware 安装 Ubuntu 22.04，请参考网络教程安装。

2. **Orange Pi 5 Plus**：

    1. 参见香橙派 5 Plus 的官方教程：Orange Pi 5 Plus。

        1. 官方用户手册位于：官方资料-> 用户手册。下载最新用户手册即可：《OrangePi_5_Plus_RK3588_用户手册_v2.1.pdf》。

        2. 下载 ubuntu 镜像：官方资料-> 官方镜像-> `Orangepi5plus_1.2.0_ubuntu_jammy_server_linux5.10.160.7z`。

            1. `server` 版本的没有桌面，因此体积更小，如有需要可以选择带桌面的版本：`Orangepi5plus_1.2.0_ubuntu_jammy_desktop_xfce_linux5.10.160.7z`。

            2. 不建议下载 `Orangepi5plus_1.2.0_ubuntu_jammy_server_linux6.1.43.7z`，WS73 驱动主要适配 Linux5.x 的内核，对于 6.x 的内核需要修改 WS73 驱动。

        3. 下载烧录工具：官方资料-> 官方工具->“Linux 镜像烧录工具-Win32DiskImager”。

    2. 根据官方教程《OrangePi_5_Plus_RK3588_用户手册_v2.1.pdf》的《2.3.1. 使用 balenaEtcher 烧录 Linux 镜像的方法》烧录镜像至 SD 卡。

    3. Linux 内核编译环境参见用户手册中《6. Linux SDK——orangepi-build 使用说明》。

3. **星闪三模网卡模块 WS73 的 SDK**：`ws73_sdk_linux_WS73_1.10.110.zip`。

4. **（可选）星闪开发板 E528H-WS63**：

    1. AT 固件及用户手册：`WS63-XF_AT-2.0.2-alpha.fwpkg` 和 `XF AT 指令使用指南-V2.0.x.pdf`。点击此处获取：《AT 固件》。

    2. 烧录工具 `BurnTool.rar`。点击此处获取：《烧录指南》。

注意事项：

1. 由于协议栈以及星闪应用海思官方不开源，需要官方编译，因此请注意以下 Linux 平台支持列表，如不在列表内需要联系海思提交工单。

```
1155, 1156, 1156_iot, 1156_ont, 3516V610, 3518_usb,
3519d, 3751, 7205_sdio, 7205_usb, 920, 9633, A40I,
rk3568, rv1126, stm32mp157, t23, t41,
```

    注 1：不同版本的 WS73 SDK 支持列表不同，此列表适用于：`WS73_1.10.110`。

    注 2：此列表可从此处获取：`ws73_sdk_linux_WS73_1.10.110/application/bin`。

2. RK3568 的程序实测可以在 RK3588 上运行。

编译 Linux 内核

请参考《OrangePi_5_Plus_RK3588_用户手册_v2.1.pdf》中的《6.1. 编译系统需求》和《6.2.获取 linux sdk 的源码》准备编译环境以及获取 Linux 源码。

笔者使用 orangepi-xunlong/orangepi-build 编译时的 SHA 为 `36a2f27f9b2d064331e4e22ccd384e0d269dbd31`。

6.2. 获取 linux sdk 的源码

6.2.1. 从 github 下载 orangepi-build

1) linux sdk 其实指的就是 orangepi-build 这套代码，orangepi-build 是基于 armbian build 编译系统修改而来的，使用 orangepi-build 可以编译出多个版本的 linux 镜像。首先下载 orangepi-build 的代码，命令如下所示：

```
test@test:~$ sudo apt-get update
test@test:~$ sudo apt-get install -y git
test@test:~$ git clone https://github.com/orangepi-xunlong/orangepi-build.git -b next
```

注意，Orange Pi 5 Plus 开发板是需要下载 orangepi-build 的 next 分支源码的，上面的 git clone 命令需要指定 orangepi-build 源码的分支为 next。

读者需要完成《6.2.获取 linux sdk 的源码》的步骤

可以忽略《6.3. 编译 u-boot》的步骤，主要参考《6.4. 编译 linux 内核》，并使用 legacy Linux 内核，默认对应 `linux5.10.160`。

接下来的步骤默认读者完成了以下操作：

1. 烧录了 `Orangepi5plus_1.2.0_ubuntu_jammy_server_linux5.10.160.7z`，并能通过串口控制。
2. 配置好了 SSH 远程连接（可以快速拷贝数据）。
3. 准备好了 Linux 内核编译环境，并且已经克隆了 orangepi-build 到虚拟机中。

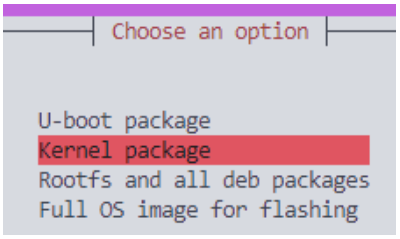
通过以下步骤为 WS73 驱动配置 RK3588 运行的 Linux 内核。

1. 首先最大化窗口（防止之后配置时因窗口太小报错），运行 build.sh 脚本。

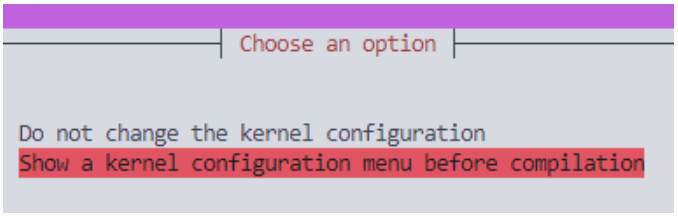
```
sudo ./build.sh
```

2. 编译 Linux 配置。

    1. 选择 `Kernel package` 后回车。

    2. 选择 `Show a kernel configuration menu before compilation` 后回车，在编译前显示内核菜单。

    3. 选择 orangepi5plus 后回车。

```
orangepi3           Allwinner H6 quad core 1GB/2GB RAM GBE WiFi/BT eMMC USB3
orangepi3-lts       Allwinner H6 quad core 2GB RAM GBE WiFi/BT-AW859A eMMC USB3
orangepizero2       Allwinner H616 quad core 512MB/1GB RAM WiFi/BT GBE SPI
orangepizero3       Allwinner H618 quad core 1GB/1.5GB/2GB/4GB RAM WiFi/BT GBE SPI
orangepizero2w      Allwinner H618 quad core 1GB/1.5GB/2GB/4GB RAM WiFi/BT SPI
orangepi4           Rockchip  RK3399 hexa core 4GB RAM GBE eMMC USB3 USB-C WiFi/BT
orangepi4a          Allwinner T527 octa core 2-4GB RAM GBE WiFi/BT NVMe eMMC
orangepi4-lts       Rockchip  RK3399 hexa core 4GB RAM GBE eMMC USB3 USB-C WiFi/BT
orangepi800         Rockchip  RK3399 hexa core 4GB RAM GBE eMMC USB3 USB-C WiFi/BT VGA
orangepi5           Rockchip  RK3588S octa core 4-16GB RAM GBE USB3 USB-C NVMe
orangepicm5         Rockchip  RK3588S octa core 4-16GB RAM GBE USB3 USB-C
orangepicm5-tablet Rockchip  RK3588S octa core 4-16GB RAM USB3 USB-C WiFi/BT
orangepi5b          Rockchip  RK3588S octa core 4-16GB RAM GBE USB3 USB-C WiFi/BT eMMC
orangepi5pro        Rockchip  RK3588S octa core 4-16GB RAM GBE USB3 WiFi/BT NVMe eMMC
orangepi5max        Rockchip  RK3588 octa core 4-16GB RAM 2.5GBE USB3 WiFi/BT NVMe eMMC
orangepi5ultra      Rockchip  RK3588 octa core 4-16GB RAM 2.5GBE USB3 WiFi/BT NVMe eMMC
orangepi5plus       Rockchip  RK3588 octa core 4-32GB RAM 2.5GBE USB3 USB-C WiFi/BT NVMe eMMC
orangepicm4         Rockchip  RK3566 quad core 2-8GB RAM GBE eMMC USB3 NvMe WiFi/BT
orangepi3b          Rockchip  RK3566 quad core 2-8GB RAM GBE eMMC USB3 NvMe WiFi/BT
orangepirv          Starfive  JH7110 quad core 2-8GB RAM GBE USB3 NvMe WiFi/BT
orangepirv2         Ky X1 octa core 2-8GB RAM GBE USB3 WiFi/BT NVMe eMMC
```

    4. 在 `Select the target kernel branch` 菜单内注意选择 `legacy  Old stable / Legacy`，这个选项才对应 `linux5.10.160`。

```
current Recommended. Come with best support
legacy  Old stable / Legacy
```

5. 如果是第一次运行，可能会在此处等待一段时间，因为要下载 Linux 内核。

```
mao@ubuntu:~/work/rk3588/orangepi-build$ sudo ./build.sh
[sudo] mao 的密码:
[ o.k. ] Using config file [ /home/mao/work/rk3588/orangepi-build/userpatches/config-default.conf ]
[ .... ] Extension being added [ rkbin-tools :: added by ./build.sh:305 -> scripts/main.sh:376 -> scripts/c
rces/families/include/rockchip64_common.inc:5 -> scripts/extensions.sh:0 ]
[ o.k. ] Extension manager [ processed 3 Extension Methods calls and 3 Extension Method implementations ]
[ o.k. ] Preparing [ host ]
[ o.k. ] Build host OS release [ jammy ]
[ .... ] Installing build dependencies
[ o.k. ] Syncing clock [ cn.pool.ntp.org ]
```

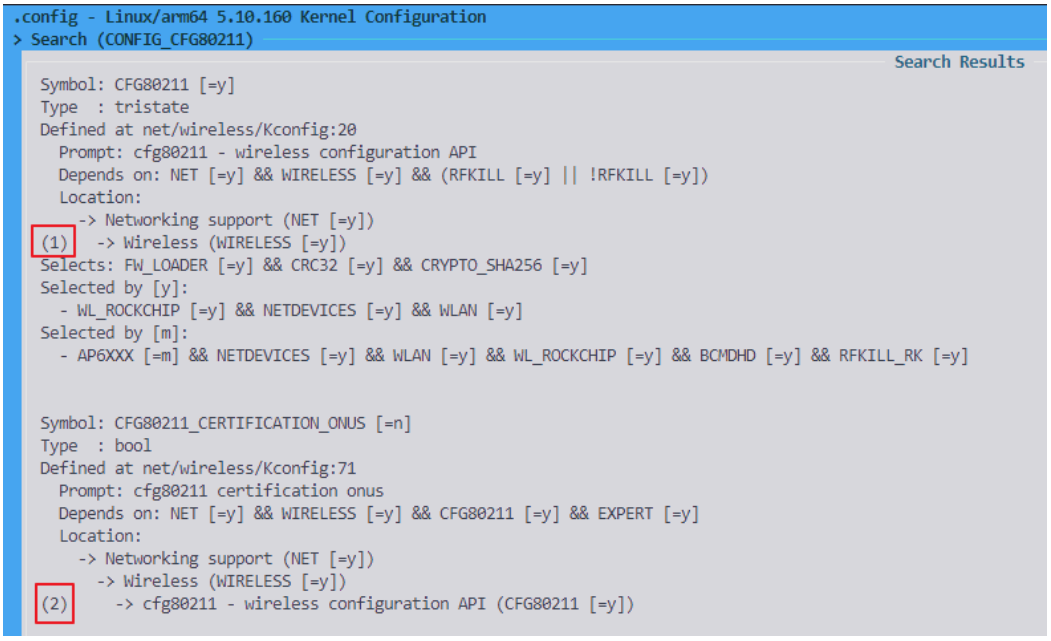
6. 之后弹出 Linux 内核配置菜单。

```
                    Linux/arm64 5.10.160 Kernel Configuration
--> (or empty submenus ----).  Highlighted letters are hotkeys.  Pressing <Y>
: [*] built-in  [ ] excluded  <M> module  < > module capable


      General setup  --->
      [*] Support DMA zone
      [*] Support DMA32 zone
          Platform selection  --->
          Kernel Features  --->
          Boot options  --->
          Power management options  --->
          CPU Power Management  --->
          Firmware Drivers  --->
      [ ] ACPI (Advanced Configuration and Power Interface) Support  ----
      [*] Virtualization  --->
      -*- ARM64 Accelerated Cryptographic Algorithms  --->
          General architecture-dependent options  --->
```

实用技巧：在 Linux 内核配置菜单界面，在英文输入法状态下，按下 `/` 就可以弹出搜索菜单（只能搜索 Kconfig 中的宏名）。

```
                    Search Configuration Parameter
  Enter (sub)string or regexp to search for (with or without "CONFIG_")


                      < Ok >         < Help >
```

输入需要搜索的宏名，比如 `CONFIG_CFG80211` 之后，会弹出搜索结果。



之后按数字键 1 或 2 就可以跳转到对应的结果。

需要关注的配置主要有以下几项（有些默认就是目标配置，不一定所有配置都要修改）。

> 也可以参考附录《附录1：Linux内核配置》直接修改配置文件，跳过下面图形化配置步骤。

1. 配置 CFG80211 和 MAC80211。
    1. **作用**：CFG80211 是内核中 Wi-Fi 驱动和用户态进程的标准接口。
    2. **宏名**：`CONFIG_CFG80211`。
    3. **位置**：`Networking support -> Wireless`。
    4. **配置**：这两项都设为 Y。

```
--- Wireless
-*-   cfg80211 - wireless configuration API
[*]     nl80211 testmode command
[ ]     enable developer warnings
[ ]     cfg80211 certification onus
[*]     enable powersave by default
[*]     cfg80211 DebugFS entries
[*]     support CRDA
[*]     cfg80211 wireless extensions compatibility
[ ]   lib80211 debugging messages
-*-   Generic IEEE 802.11 Networking Stack (mac80211)
[*]   Minstrel
      Default rate control algorithm (Minstrel)  --->
[ ]   Enable mac80211 mesh networking support
-*-   Enable LED triggers
[*]   Export mac80211 internals in DebugFS
[ ]   Trace all mac80211 debug messages
[*]   Select mac80211 debugging features  --->
```

如果设为'M'，在后续步骤插入驱动前要首先插入 `cfg80211.ko`。

2. （使用 SDIO 接口的 M512H-WS73 才要配置）配置 SDIO（此配置默认使能）。
    1. **宏名**：`CONFIG_MMC`。
    2. **位置**：`Device Drivers`。
    3. **配置**：将 `MMC/SD/SDIO card support` 设为 Y。

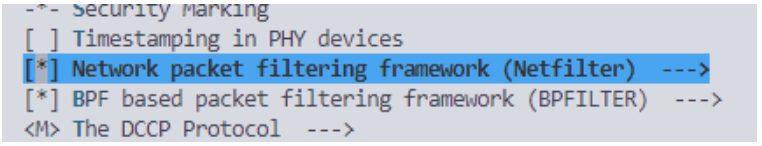
```
    HID support  --->
[*] USB support  --->
<*> MMC/SD/SDIO card support  --->
< > Sony MemoryStick card support  ----
[*] LED Support  --->
```

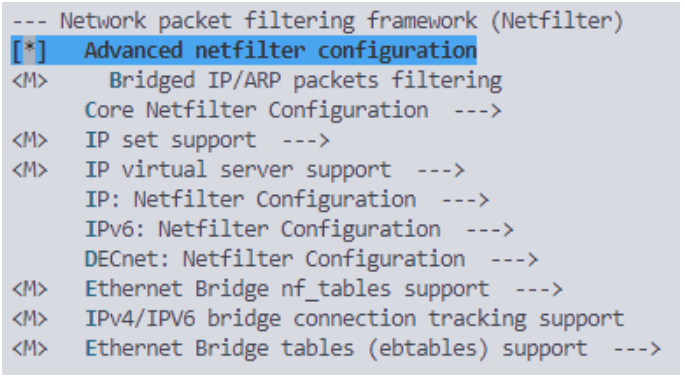
3. （使用 USB 接口的 M573H-USBDG 才要配置）配置 USB（此配置默认使能）。
    1. **宏名**：`CONFIG_USB_XHCI_HCD`。
    2. **位置**：`Device Drivers -> USB support`。
    3. **配置**：将 `xHCI HCD (USB 3.0) support` 设为 Y。

```
<M>   USB Monitor
      *** USB Host Controller Drivers ***
< >   Cypress C67x00 HCD support
<*>   xHCI HCD (USB 3.0) support
[ ]     xHCI support for debug capability
< >     Support for additional Renesas xHCI
-*-     Generic xHCI driver for a platform
<*>   EHCI HCD (USB 2.0) support
```

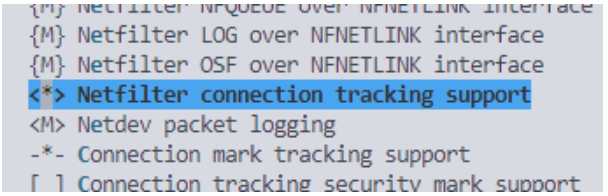
4. 配置 Netlink。

1. **作用**：由于 wpa_supplicant、hostapd 应用采用 Netlink 技术与 Linux 内核通信，需要配置 Netlink。
2. **宏名**：`CONFIG_NETFILTER` 和 `CONFIG_NETFILTER_ADVANCED`。
3. **位置**：`Networking support -> Networking options`。
4. **配置**：将 `Network packet filtering framework (Netfilter)` 设为 Y 后，进入并设置 `Advanced netfilter Configuration` 为 Y。





5. （可选）配置 NAT 转发。
    1. **作用**：如果需要使用 SoftAP 网络共享功能，需要配置 NAT 转发功能。
    2. **步骤 1**：
        1. **宏名**：`CONFIG_NF_CONNTRACK`。
        2. **位置**：`Networking support -> Networking options -> Network packet filtering framework (Netfilter) -> Core Netfilter Configuration`
        3. **配置**：`Netfilter connection tracking support` 设为 Y。



    3. **步骤 2**：
        1. **宏名**：`CONFIG_IP_NF_ARPTABLES`。

2. **位置**：`Networking support -> Networking options -> Network packet filtering framework (Netfilter) -> IP: Netfilter Configuration`
3. **配置**：`ARP tables support` 设为 Y。

```
<M>     TTL target support
<M>   raw table support (required for NOTRACK/TRACE)
<M>   Security table
<*> ARP tables support
<M>   ARP packet filtering
<M>   ARP payload mangling
```

6. 配置 BT。
   1. **作用**：如果需要使用 BT 功能，需要开启蓝牙内核模块编译选项。
   2. **宏名**：`CONFIG_BT` 。
   3. **位置**：`Networking support`。
   4. **配置**：将 `Bluetooth subsystem support` 设为 Y。

```
--- Networking support
      Networking options  --->
[ ]   Amateur Radio support  ----
<M>   CAN bus subsystem support  --->
<*>   Bluetooth subsystem support  --->
{M}   RxRPC session sockets
```

7. 按 S 保存配置，推出菜单后开始编译。初次编译时间根据电脑配置而不同，几分钟到几十分钟。

```
mao@ubuntu:~/work/rk3588/orangepi-build$ sudo ./build.sh
[sudo] mao 的密码:
[ o.k. ] Using config file [ /home/mao/work/rk3588/orangepi-build/userpatches/config-default.conf ]
[ .... ] Extension being added [ rkbin-tools :: added by ./build.sh:305 -> scripts/main.sh:376 -> scripts/configuration.sh:151 -> ex
rces/families/include/rockchip64_common.inc:5 -> scripts/extensions.sh:0 ]
[ o.k. ] Extension manager [ processed 3 Extension Methods calls and 3 Extension Method implementations ]
[ o.k. ] Preparing [ host ]
[ o.k. ] Build host OS release [ jammy ]
[ .... ] Installing build dependencies
[ o.k. ] Syncing clock [ cn.pool.ntp.org ]
Exiting, name server cannot be used: Temporary failure in name resolution (-3)[ o.k. ] Checking for external GCC compilers
[ o.k. ] Cleaning /home/mao/work/rk3588/orangepi-build/output/debs for [ orangepi5plus legacy ]
[ o.k. ] Started patching process for [ kernel rockchip-rk3588-legacy ]
[ o.k. ] Looking for user patches in [ userpatches/kernel/rockchip-rk3588-legacy ]
[ o.k. ] Compiling legacy kernel [ 5.10.160 ]
[ o.k. ] Compiler version [ aarch64-none-linux-gnu-gcc 11.2.1 ]
[ o.k. ] Using kernel config file [ /home/mao/work/rk3588/orangepi-build/external/config/kernel/linux-rockchip-rk3588-legacy.config
#
# No change to .config
#


*** End of the configuration.
*** Execute 'make' to start the build or try 'make help'.

[ o.k. ] Exporting new kernel config [ /home/mao/work/rk3588/orangepi-build/output/config/linux-rockchip-rk3588-legacy.config ]
  SYNC    include/config/auto.conf.cmd
  CALL    scripts/atomic/check-atomics.sh
  CALL    scripts/checksyscalls.sh
  CHK     include/generated/compile.h
  GEN     .version
  CHK     include/generated/compile.h
  UPD     include/generated/compile.h
  CC      init/version.o
  AR      init/built-in.a
  LD      vmlinux.o
  MODPOST vmlinux.symvers
  MODINFO modules.builtin.modinfo
  GEN     modules.builtin
  LD      .tmp_vmlinux.kallsyms1
  KSYMS   .tmp_vmlinux.kallsyms1.S
  AS      .tmp_vmlinux.kallsyms1.S
  LD      .tmp_vmlinux.kallsyms2
  KSYMS   .tmp_vmlinux.kallsyms2.S
  AS      .tmp_vmlinux.kallsyms2.S
  LD      vmlinux
```

8. 编译完毕后，生成的内核相关的 deb 包位于 `orangepi-build/output/debs` 目录内。

```
  HOSTCC  scripts/unifdef
  INSTALL ./debian/headertmp/usr/include
  CLEAN   scripts/basic
  CLEAN   scripts/genksyms
  CLEAN   scripts/kconfig
  CLEAN   scripts/mod
  CLEAN   scripts/selinux/genheaders
  CLEAN   scripts/selinux/mdp
  CLEAN   scripts/dtc
  CLEAN   scripts
patching file tools/include/tools/be_byteshift.h
patching file tools/include/tools/le_byteshift.h
dpkg-deb: 正在 '../linux-headers-legacy-rockchip-rk3588_1.2.0_arm64.deb' 中构建软件包 'linux-headers-legacy-rockchip-rk3588'。
dpkg-deb: 正在 '../linux-dtb-legacy-rockchip-rk3588_1.2.0_arm64.deb' 中构建软件包 'linux-dtb-legacy-rockchip-rk3588'。
dpkg-deb: 正在 '../linux-image-legacy-rockchip-rk3588_1.2.0_arm64.deb' 中构建软件包 'linux-image-legacy-rockchip-rk3588'。
dpkg-genchanges: info: binary-only upload (no source code included)
dpkg-buildpackage: info: binary-only upload (no source included)
[ o.k. ] Kernel build done [ @host ]
[ o.k. ] Target directory [ /home/mao/work/rk3588/orangepi-build/output/debs/ ]
[ o.k. ] File name [ linux-image-legacy-rockchip-rk3588_1.2.0_arm64.deb ]
[ o.k. ] Runtime [ 51 min ]
[ o.k. ] Repeat Build Options [ sudo ./build.sh  BOARD=orangepi5plus BRANCH=legacy BUILD_OPT=kernel KERNEL_CONFIGURE=yes  ]
```

```
debs/
├ linux-dtb-legacy-rockchip-rk3588_1.2.0_arm64.deb       # 包含内核使用的 dtb 文件
├ linux-headers-legacy-rockchip-rk3588_1.2.0_arm64.deb   # 包含内核头文件

```

```
└ linux-image-legacy-rockchip-rk3588_1.2.0_arm64.deb    # 包含内核镜像和内核模块
```

其中需要的是：`linux-image-legacy-rockchip-rk3588_1.2.0_arm64.deb`。

接下来，在 PC 中通过以下步骤更新开发板 Linux 系统的内核和内核模块。

```
# 在 orangepi-build 目录下移动到 debs 目录内
$ cd output/debs
# 通过 scp 复制到开发板
# 其中 `192.168.43.30` 是笔者开发板的 IP 地址
$ scp linux-image-legacy-rockchip-rk3588_1.2.0_arm64.deb orangepi@192.168.43.30:/home/orangepi
```

在开发板中通过以下步骤更新新的 Linux 内核的 deb 包：

```
$ cd /home/orangepi
# 卸载已安装的linux内核的deb包
$ sudo apt purge-y linux-image-legacy-rockchip-rk3588
# 安装刚才上传的新的linux内核的deb包
$ sudo dpkg-i linux-image-legacy-rockchip-rk3588_1.2.0_arm64.deb
# 重启，让新的内核生效
$ reboot
```

Linux 平台驱动移植

通过《编译 Linux 内核》的步骤，已经为 WS73 修改了 RK3588 的 Ubuntu 镜像的 Linux 内核。接下来为 WS73 编译驱动。

以下步骤在 PC 端操作：

1. 复制 `ws73_sdk_linux_WS73_1.10.110.zip` 到虚拟机中，并解压 WS73 SDK。

```
mao@ubuntu:~/nfs_server/ws73$ ls
ws73_sdk_linux_WS73_1.10.110.zip
mao@ubuntu:~/nfs_server/ws73$ unzip ws73_sdk_linux_WS73_1.10.110.zip
Archive:  ws73_sdk_linux_WS73_1.10.110.zip
   creating: ws73_sdk_linux_WS73_1.10.110/
  inflating: ws73_sdk_linux_WS73_1.10.110/Makefile
```

```
$ unzip ws73_sdk_linux_WS73_1.10.110.zip
```

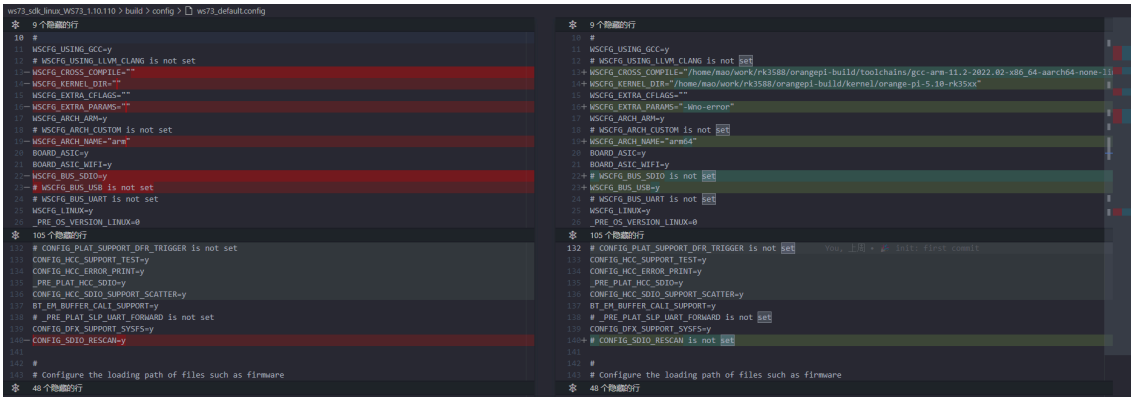
2. 修改默认 config 配置。

可以参考《附录2：WS73参考配置》的 diff 修改。

默认编译配置位于 `ws73_sdk_linux_WS73_1.10.110/build/config/ws73_default.config` 。

需要关注和修改的配置如下：

参数	说明	示例
WSCFG_USING_GCC	是否配置编译器类型为 GCC	WSCFG_USING_GCC=y
WSCFG_CROSS_COMPILE	配置编译器存放路径	（注意修改 orangepi-build 到实际路径） WSCFG_CROSS_COMPILE="/home/mao/work/rk3588/orangepi-build/toolchains/gcc-arm-11.2-2022.02-x86_64-aarch64-none-linux-gnu/bin/aarch64-none-linux-gnu-"
WSCFG_KERNEL_DIR	配置内核存放路径	（注意修改 orangepi-build 到实际路径） WSCFG_KERNEL_DIR="/home/mao/work/rk3588/orangepi-build/kernel/orange-pi-5.10-rk35xx"
WSCFG_EXTRA_PARAMS	配置额外的编译参数	WSCFG_EXTRA_PARAMS="-Wno-error"
WSCFG_ARCH_ARM	配置 CPU 架构类型是否为 ARM	WSCFG_ARCH_ARM=y
WSCFG_ARCH_NAME	配置 CPU 架构名	（对于 RK3588 需要使用 arm64） WSCFG_ARCH_NAME="arm64"
WSCFG_BUS_SDIO	配置 WS73 网卡是否使用 SDIO 接口，和 WSCFG_BUS_USB 之间二选一	对于 **M573H-USBDG**： # WSCFG_BUS_SDIO is not set 对于 **M512H-WS73**： WSCFG_BUS_SDIO=y
WSCFG_BUS_USB	配置 WS73 网卡是否使用 USB 接口，和 WSCFG_BUS_SDIO 之间二选一	对于 **M573H-USBDG**： WSCFG_BUS_USB=y 对于 **M512H-WS73**： # WSCFG_BUS_USB is not set



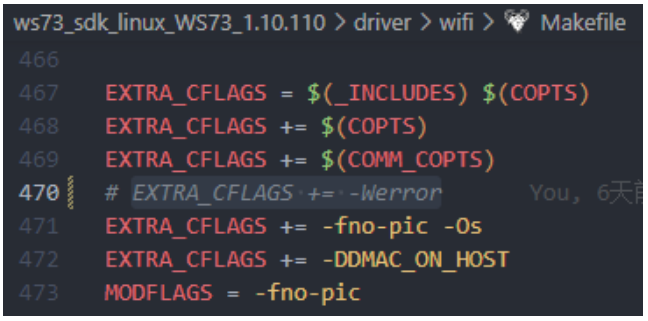
参考修改

3. 修改编译脚本。

    1. `ws73_sdk_linux_WS73_1.10.110/driver/wifi/Makefile`

        注释掉 470 行的 `EXTRA_CFLAGS += -Werror`，以避免影响编译。

```
diff --git a/driver/wifi/Makefile b/driver/wifi/Makefile
index dc84368..8e90d32 100644
--- a/driver/wifi/Makefile
+++ b/driver/wifi/Makefile
@@ -467,7 +467,7 @@ endif
 EXTRA_CFLAGS = $(_INCLUDES) $(COPTS)
 EXTRA_CFLAGS += $(COPTS)
 EXTRA_CFLAGS += $(COMM_COPTS)
-EXTRA_CFLAGS += -Werror
+# EXTRA_CFLAGS += -Werror
 EXTRA_CFLAGS += -fno-pic -Os
 EXTRA_CFLAGS += -DDMAC_ON_HOST
 MODFLAGS = -fno-pic
```



    2. `ws73_sdk_linux_WS73_1.10.110/driver/platform/Makefile`

        删除 307 行的 `-Werror`，以避免影响编译。

```
diff --git a/driver/platform/Makefile b/driver/platform/Makefile
index c0f36bb..8387ed1 100644
--- a/driver/platform/Makefile
+++ b/driver/platform/Makefile
@@ -304,7 +304,7 @@ EXTRA_CFLAGS += $(COPTS) $(KBUILD_CFLAGS)
 EXTRA_CFLAGS += $(HCC_EXTRA_CFLAGS)

 # CleanCode Build Option
-EXTRA_CFLAGS += -Werror -Os
+EXTRA_CFLAGS += -Os

```

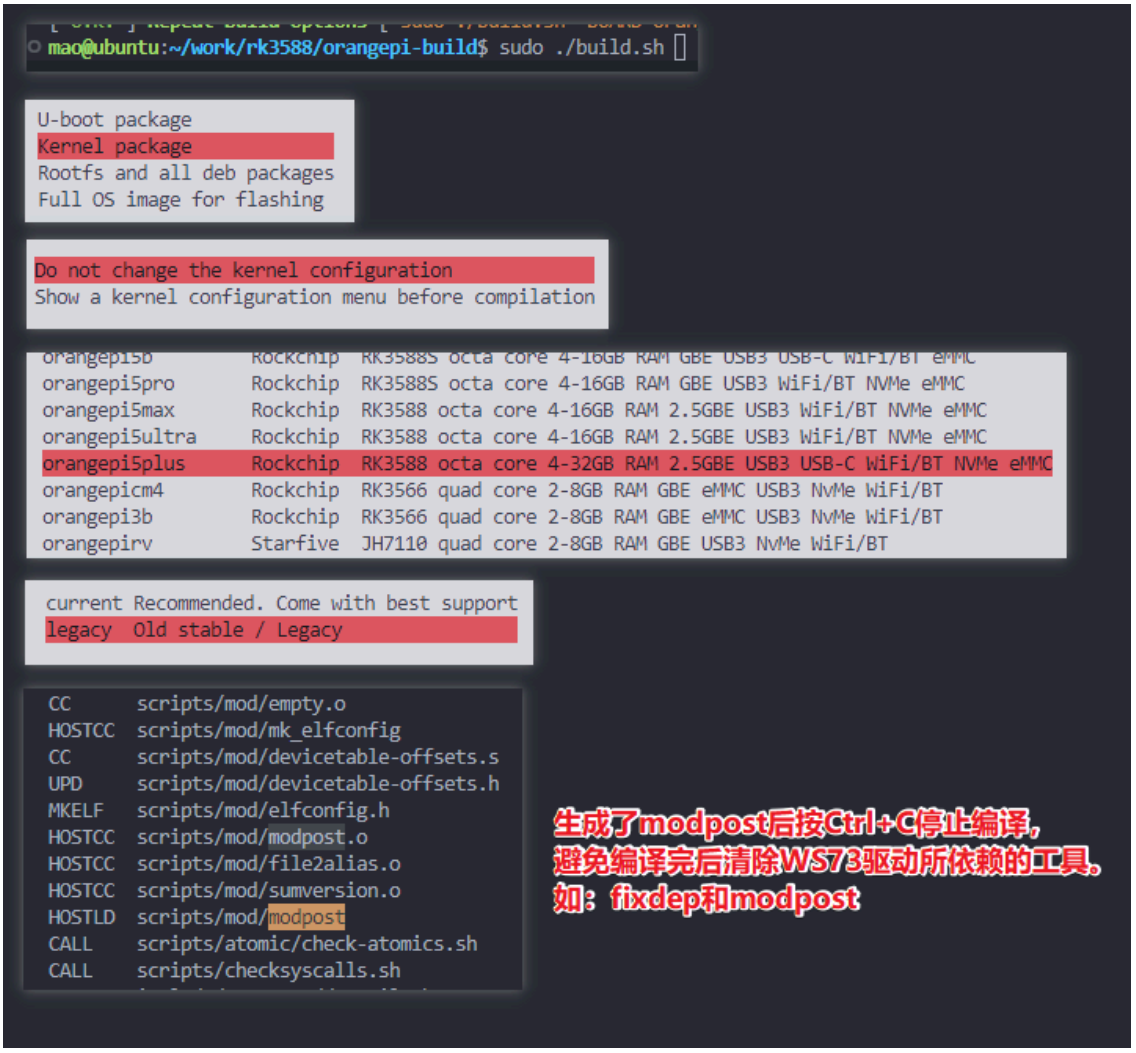
```
12  ifeq ($(BOARD_ASIC),y)
13  EXTRA_CFLAGS += -DCHIP_ASIC=1
```

ws73_sdk_linux_WS73_1.10.110 > driver > platform > Makefile

```
304  304  EXTRA_CFLAGS += $(HCC_EXTRA_CFLAGS)
305  305
306  306  # CleanCode Build Option
307       - EXTRA_CFLAGS += -Werror -Os
     307  + EXTRA_CFLAGS += -Os
308  308
309  309  ifeq ($(BOARD_ASIC),y)
310  310  EXTRA_CFLAGS += -DCHIP_ASIC=1
```

4. 生成依赖。

WS73 依赖 `orangepi-build` 中 Linux 内核编译时的一些工具，而 `orangepi-build` 的 `orange-pi-5.10-rk35xx` 编译完毕后又会删除这些工具，最简单的办法是在 `orangepi-build` 内运行完 `sudo ./build.sh`，生成 `modpost` 后停止编译，直接使用生成的工具。



5. 编译。

   在虚拟机中，移动到刚才解压的 `ws73_sdk_linux_WS73_1.10.110` 根目录，通过 `make all` 命令编译 WS73 驱动。

```
mao@ubuntu:~/nfs_server/ws73_sdk_linux_WS73_1.10.110$ make all
the item protocol_enable=0 in ini file isnot in the autoconfig.h file.
the item ble_disable_ll_privacy=0 in ini file isnot in the autoconfig.h file.
the item bsle_front_switch=0 in ini file isnot in the autoconfig.h file.
the item bsle_suspend_mode=0 in ini file isnot in the autoconfig.h file.
INI file generate success in /home/mao/nfs_server/ws73_sdk_linux_WS73_1.10.110/output/bin/ws73_cfg.ini!
cd /home/mao/nfs_server/ws73_sdk_linux_WS73_1.10.110/driver/platform && \
    BUILD_DEVICE_WITH_ROM_REPO=yes\
    make -j16 && cp -f plat_soc.ko /home/mao/nfs_server/ws73_sdk_linux_WS73_1.10.110/output/bin/
make[1]: 进入目录“/home/mao/nfs_server/ws73_sdk_linux_WS73_1.10.110/driver/platform”
make -C "/home/mao/work/rk3588/orangepi-build/kernel/orange-pi-5.10-rk35xx" ARCH="arm64" CROSS_COMPILE=/h
4-aarch64-none-linux-gnu/bin/aarch64-none-linux-gnu- M=/home/mao/nfs_server/ws73_sdk_linux_WS73_1.10.110/
        -Wno-error
make[2]: 进入目录“/home/mao/nfs_server/ws73_sdk_linux_WS73_1.10.110/driver/platform”
make[2]: 警告：子 make 中强制 -j0: 重置 jobserver 模式。
```

```
WARNING: modpost: "pm_sle_resume_cb_host_register" [/home/mao/nfs_server/ws73_sdk_linux_WS73_1.10.11
WARNING: modpost: "osal_kfree_ws73" [/home/mao/nfs_server/ws73_sdk_linux_WS73_1.10.110/driver/bsle/s
WARNING: modpost: "hcc_switch_status" [/home/mao/nfs_server/ws73_sdk_linux_WS73_1.10.110/driver/bsle
  CC [M]  /home/mao/nfs_server/ws73_sdk_linux_WS73_1.10.110/driver/bsle/sle_driver/sle_soc.mod.o
  LD [M]  /home/mao/nfs_server/ws73_sdk_linux_WS73_1.10.110/driver/bsle/sle_driver/sle_soc.ko
make[2]: 离开目录“/home/mao/work/rk3588/orangepi-build/kernel/orange-pi-5.10-rk35xx”
"/home/mao/work/rk3588/orangepi-build/toolchains/gcc-arm-11.2-2022.02-x86_64-aarch64-none-linux-gnu/
make[1]: 离开目录“/home/mao/nfs_server/ws73_sdk_linux_WS73_1.10.110/driver/bsle/sle_driver”
sle ko built success in /home/mao/nfs_server/ws73_sdk_linux_WS73_1.10.110/output/bin/sle_soc.ko!
all built!
```

   编译完毕

6. 编译完毕后，对于本文使用的 USB 接口的 **M573H-USBDG**，需要修改 `ws73_sdk_linux_WS73_1.10.110/output/bin/ws73_cfg.ini` 中的 `power_gpio_idx` 为-1。

   这是因为 USB 接口的 M573H-USBDG 没有 GPIO 控制 WS73 芯片上下电/复位。

```
227  227  # gpio序号配置，配为-1代表不支持
228  228  # power_on管脚，power_on_level为使73power_on的电平，支持配0-低电平/1-高电平，
229       - power_gpio_idx=40
     229+ power_gpio_idx=-1
```

7. 编译结果。

   编译结果输出到"output/bin"目录下，如下所示：

```
bin/
├ autoconfig.h
├ ble_soc.ko      # 蓝牙驱动模块
├ bp_channel
├ bp_test.ko
├ plat_soc.ko     # 平台驱动模块
├ sle_soc.ko      # 星闪驱动模块
```

```
├ wifi_soc.ko     # WiFi驱动模块
└ ws73_cfg.ini    # 客制化的配置文件
```

8. 拷贝所需文件。

   通过 scp 等方式拷贝 WS73 所需的文件到开发板对应目录，如下所示：

SDK 内的文件	开发板存放目录
`firmware/e/` 或 `firmware/us/` 内的**所有文件**，需要根据实际芯片选择，笔者使用 e。	/etc/ws73/（需要用户创建）
修改后的 output/bin/ws73_cfg.ini	/etc/
`output/bin` 的 ble_soc.ko, plat_soc.ko, sle_soc.ko, wifi_soc.ko	任意目录，如：/komod/

   对于不同功能还需额外拷贝不同功能所需的文件，下文将会提及。

测试

WS73 的星闪功能测试

测试 WS73 的星闪功能还需由海思官方编译的星闪 AT 程序，支持的 Linux 平台已经在前文的《Linux 平台支持列表》给出。经过测试 RK3588 可以使用 RK3568 的程序，此处以 RK3568 的程序为例演示如何使用海思官方编译的星闪 AT 程序。

1. 拷贝所需的文件。

   1. 拷贝测试 WS73 的星闪功能需要 `ws73_sdk_linux_WS73_1.10.110/application/bin/<你的Linux平台>/sle/` 的 `sparklinkd` 和 `sparklinkctrl` 到开发板任意目录下，并修改为可执行权限。

      在 PC 内执行以下命令：

      ```
      $ cd ws73_sdk_linux_WS73_1.10.110/application/bin/rk3568/sle/
      # 其中 `192.168.43.30` 是笔者开发板的 IP 地址，目标路径根据实际情况修改
      $ scp sparklinkd sparklinkctrl
      orangepi@192.168.43.30:/home/orangepi/work/ws73_app/sle
      ```

      在香橙派 5 Plus 开发板内执行以下命令：

```
$ cd /home/orangepi/work/ws73_app/sle
$ sudo chmod u+x sparklinkctrl sparklinkd
```

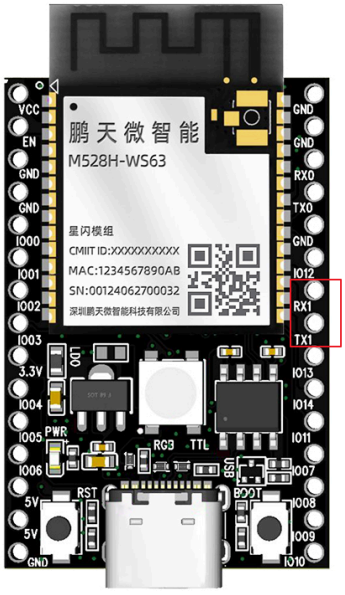
（根据实际情况选择）WS73 与 WS63 星闪功能测试

请注意，目前 WS63 的 SDK （**1.10.102** 及之前）的星闪广播包格式还未更新，而 WS73 的 SDK （**1.10.110**）更新了星闪广播包格式，因此只有 WS73 能搜到 WS63 的广播包，而 WS63 因为无法识别 WS73 的广播包而无法扫描和连接。

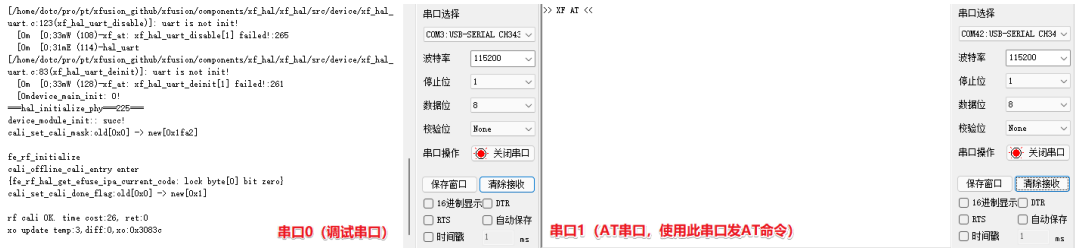
通过以下步骤完成 WS73 与 WS63 星闪功能测试。

> 其中 WS63 作为服务器广播并接受连接，WS73 作为客户端，扫描 WS63 的广播，并连接 WS63。

1. 烧录 WS63 AT 固件。
    1. WS63 AT 固件和烧录指南可通过《AT 固件》和《烧录指南》获取，请根据烧录指南烧录 `WS63-XF_AT-2.0.2-alpha.fwpkg` 固件。
    2. 由于 AT 固件默认使用串口 1，请接连接串口 1 后（注意 TX 和 RX 交叉连接，不要忘记 GND）使用串口 1 与 WS63 交互。



    3. 烧录完后后的效果如图所示。后续使用输出 `>> XF AT <<` 的串口发送 AT 命令



2. 在 **WS63** 上启动服务端并开启广播。

   在 WS63 的在 AT 串口上输入以下命令，启动服务端并开启广播，命令详情请见《`XF AT 指令使用指南-V2.0.x.pdf`》

```
AT+ECHO=1
AT+SLEENABLE=1
AT+SLEOBJ=1,0
AT+SLEADDR=0
AT+SLEADV=0,1
```

```
>> XF AT <<

OK
AT+SLEENABLE=1

OK
AT+SLEOBJ=1,0
ID:0

OK
AT+SLEADDR=0
0,CA:CA:CA:CA:CA:CA

OK
AT+SLEADV=0,1

OK
```

   运行结果

   以上命令完成之后，WS63 启动服务端并开启广播。注意 WS63 的 MAC 地址默认是`cacacacacaca`，下文需要用到。

3. 在**香橙派 5 Plus** 上驱动 WS73。

   > NOTE：在插入 WS73 网卡前，必须完成《Linux 平台驱动移植》的`8. 拷贝所需文件。`的步骤。

   1. 插入并检测 WS73 网卡。

      1. 首先插入 WS73 网卡到香橙派 5 Plus。

2. 对于 **M573H-USBDG**，需要在 Host 端检测到 USB 设备：

    1. 如果主控平台支持 USB 功能且是 build-in 模式，则 Linux 启动时会检测到 USB 设备，显示“xHCI Host Controller”打印信息，说明探测到 USB 设备。
    2. 如果主控平台支持 USB 功能且需要加载驱动，则需要加载 USB 驱动后，再加载 UB73 驱动。
    3. 通过 `lsusb` 查看是否检测到 USB 设备，模组的默认 ID 为 `ffff:3733`，如下图所示。

    ```
    orangepi@orangepi5plus:~$ lsusb
    Bus 006 Device 002: ID 05e3:0620 Genesys Logic, Inc. USB3.2 Hub
    Bus 006 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
    Bus 005 Device 002: ID 05e3:0610 Genesys Logic, Inc. Hub
    Bus 005 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
    Bus 008 Device 001: ID 1d6b:0003 Linux Foundation 3.0 root hub
    Bus 007 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
    Bus 004 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
    Bus 002 Device 036: ID ffff:3733 00000000 00000000
    Bus 002 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
    Bus 003 Device 001: ID 1d6b:0001 Linux Foundation 1.1 root hub
    Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
    ```

3. 对于 **M512H-WS73**，需要在 Host 端检测到 SDIO 设备：

    1. 如果芯片是默认上电的，则 Linux 启动时会检测到 SDIO 设备，“mmc1:newSDIO card at address 0001”打印信息，说明 SDIO 检测成功。
    2. 如果芯片是默认不上电的，则需要在 WiFi 驱动中添加上电 SDIO 检测相关流程，再继续执行。
    3. 平台可以通过执行 `cat /proc/mci/mci_info` 命令查看是否检测到 SDIO 设备。

2. 加载驱动。

```
# 移动到之前拷贝ko到开发板的目录，根据实际情况配置
$ cd /home/orangepi/work/ws73_komod
# 首先加载平台驱动模块plat_soc.ko，再插入星闪驱动模块sle_soc.ko
$ sudo insmod plat_soc.ko
$ sudo insmod sle_soc.ko
```

```
orangepi@orangepi5plus:~/work/ws73_komod$ ls
ble_soc.ko  plat_soc.ko  sle_soc.ko  wifi_soc.ko
orangepi@orangepi5plus:~/work/ws73_komod$ sudo insmod plat_soc.ko
[sudo] password for orangepi:
orangepi@orangepi5plus:~/work/ws73_komod$ sudo insmod sle_soc.ko
```

成功加载的情况下，是没有任何其他信息输出的。

> NOTE：如果想要取消加载，可通过以下命令实现：

```
$ sudo rmmod sle_soc
$ sudo rmmod plat_soc
```

3. 运行 WS73 的 AT 程序。

```
# 移动到《WS73 的星闪功能测试》拷贝app到开发板的目录，根据实际情况配置
$ cd /home/orangepi/work/sle
$ ./sparklinkd &
$ ./sparklinkctrl
```

```
orangepi@orangepi5plus:~/work/ws73_app/sle$ ls
sparklinkctrl  sparklinkd
orangepi@orangepi5plus:~/work/ws73_app/sle$ ./sparklinkd &
[1] 23966
spark link main start.

orangepi@orangepi5plus:~/work/ws73_app/sle$ ./sparklinkctrl
waiting server start...
server started
sparklinkctrl>
```

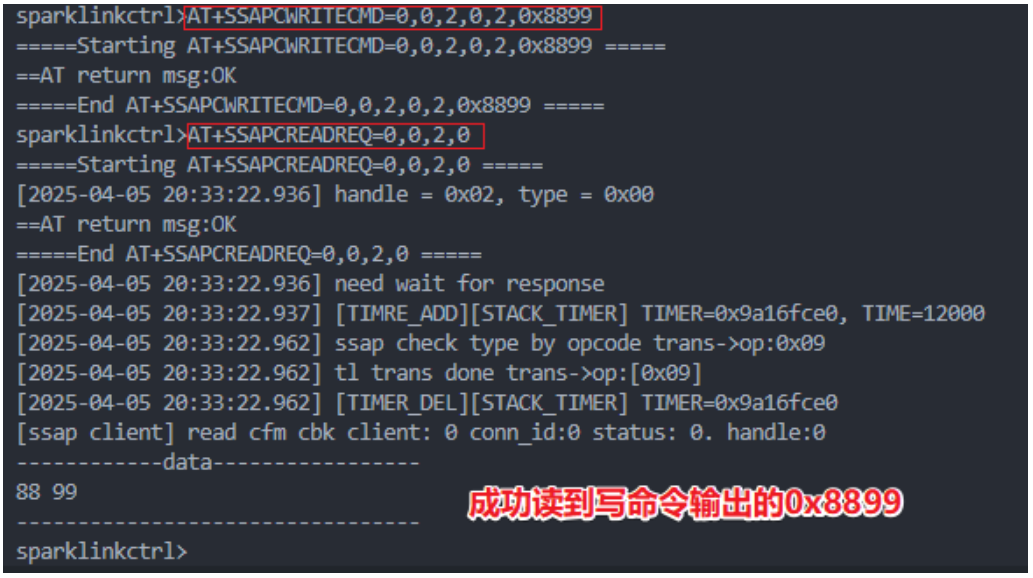
之后，WS73的AT命令行开始运行

> NOTE：sparklinkd 在后台运行，如果想要结束 sparklinkd，可以通过此命令结束：`pkill -9 -f sparklinkd`。

4. 在 `sparklinkctrl` 的命令行内输入以下 AT 命令，连接 WS63 服务器，发送数据后读取数据。

> NOTE：`sparklinkctrl` 的 AT 命令是官方 AT 固件，因此和 WS63 的 AT 命令不一致。

```
# 使能SLE
AT+SLEENABLE
# 获取本机地址
AT+SLEGETADDR
# 设置扫描参数
AT+SLESETSCANPAR=0,0x48,0x48
# 开始扫描，注意，扫描开始之后不会主动停止，需要输入后面的停止命令才能停止
AT+SLESTARTSCAN
# 停止扫描
AT+SLESTOPSCAN
# 注册 SSAPC 回调
AT+SSAPCREGCBK
# 连接对端，注意此处 0xcacacacacaca 为服务端的地址
AT+SLECONN=0,0xcacacacacaca
# 通过uuid发现服务
AT+SSAPCFNDSTRU=0,0,1
# 发送数据0x8899
AT+SSAPCWRITECMD=0,0,2,0,2,0x8899
# 读取服务端属性数据
AT+SSAPCREADREQ=0,0,2,0
```



正常运行

5. 测试完毕。

（根据实际情况选择）双 WS73 星闪功能测试

这种情况要求准备**两**个**香橙派 5 Plus** 和**两**个 WS73 网卡，操作步骤与《（根据实际情况选择）WS73 与 WS63 星闪功能测试》类似。

1. 在两一个香橙派 5 Plus 分别执行《《（根据实际情况选择）WS73 与 WS63 星闪功能测试》》中的《在香橙派 5 Plus 上驱动 WS73。》，启动完毕 `sparklinkctrl` 后依照以下步骤操作。
2. 在一个香橙派 5 Plus 的 AT 命令行终端中执行以下命令，启动服务器后开广播。

```
# 使能SLE
AT+SLEENABLE
# 设置本机地址
AT+SLESETADDR=0,0x012233445566
# 注册服务
AT+SSAPSADDSRV=0x1234
# 添加服务
AT+SSAPSADDSERV=0x2222,1
# 添加属性
AT+SSAPSADDPROPERTY=1,0x2323,5,5,2,0x1234
# 添加属性描述
AT+SSAPSADDDESCR=1,2,0x3333,5,5,2,2,0x0200
# 启动服务
AT+SSAPSSTARTSERV=1
# 注册服务回调
AT+SSAPSREGCBK
```

```
# 设置广播参数
AT+SLESETADVPAR=1,3,200,200,0,0x012233445566,0,000000000000
# 设置广播数据
AT+SLESETADVDATA=1,10,4,aabbccddeeff11223344,11224455
# 开启广播
AT+SLESTARTADV=1
```

3. 在另一个香橙派 5 Plus 的 AT 命令行终端中执行以下命令，连接服务器，发送数据后读取数据。

```
# 使能SLE
AT+SLEENABLE
# 获取本机地址
AT+SLEGETADDR
# 设置扫描参数
AT+SLESETSCANPAR=0,0x48,0x48
# 开始扫描
AT+SLESTARTSCAN
# 停止扫描
AT+SLESTOPSCAN
# 注册 SSAPC 回调
AT+SSAPCREGCBK
# 连接对端
AT+SLECONN=0,0x012233445566
# 通过uuid发现服务
AT+SSAPCFNDSTRU=0,0,1
# 发送数据0x8899
AT+SSAPCWRITECMD=0,0,2,0,2,0x8899
# 读取服务端属性数据
AT+SSAPCREADREQ=0,0,2,0
```

4. 测试完毕。

附录

附录 1：Linux 内核配置

```
diff --git a/external/config/kernel/linux-rockchip-rk3588-legacy.config
b/external/config/kernel/linux-rockchip-rk3588-legacy.config
index 68a0a34..c909f79 100644
--- a/external/config/kernel/linux-rockchip-rk3588-legacy.config
+++ b/external/config/kernel/linux-rockchip-rk3588-legacy.config
@@ -1113,7 +1113,7 @@ CONFIG_NETFILTER_NETLINK_ACCT=m
```

```
 CONFIG_NETFILTER_NETLINK_QUEUE=m
 CONFIG_NETFILTER_NETLINK_LOG=m
 CONFIG_NETFILTER_NETLINK_OSF=m
-CONFIG_NF_CONNTRACK=m
+CONFIG_NF_CONNTRACK=y
 CONFIG_NF_LOG_COMMON=m
 CONFIG_NF_LOG_NETDEV=m
 CONFIG_NETFILTER_CONNCOUNT=m
@@ -1348,7 +1348,7 @@ CONFIG_IP_VS_PE_SIP=m
 #
 # IP: Netfilter Configuration
 #
-CONFIG_NF_DEFRAG_IPV4=m
+CONFIG_NF_DEFRAG_IPV4=y
 CONFIG_NF_SOCKET_IPV4=m
 CONFIG_NF_TPROXY_IPV4=m
 CONFIG_NF_TABLES_IPV4=y
@@ -1382,7 +1382,7 @@ CONFIG_IP_NF_TARGET_ECN=m
 CONFIG_IP_NF_TARGET_TTL=m
 CONFIG_IP_NF_RAW=m
 CONFIG_IP_NF_SECURITY=m
-CONFIG_IP_NF_ARPTABLES=m
+CONFIG_IP_NF_ARPTABLES=y
 CONFIG_IP_NF_ARPFILTER=m
 CONFIG_IP_NF_ARP_MANGLE=m
 # end of IP: Netfilter Configuration
@@ -1423,7 +1423,7 @@ CONFIG_IP6_NF_TARGET_MASQUERADE=m
 CONFIG_IP6_NF_TARGET_NPT=m
 # end of IPv6: Netfilter Configuration

-CONFIG_NF_DEFRAG_IPV6=m
+CONFIG_NF_DEFRAG_IPV6=y

 #
 # DECnet: Netfilter Configuration
@@ -1758,7 +1758,7 @@ CONFIG_CAN_PEAK_USB=m
 CONFIG_CAN_DEBUG_DEVICES=y
 # end of CAN Device Drivers

-CONFIG_BT=m
+CONFIG_BT=y
 CONFIG_BT_BREDR=y
 CONFIG_BT_RFCOMM=m
 CONFIG_BT_RFCOMM_TTY=y
```

附录 2：WS73 参考配置

```
diff --git a/build/config/ws73_default.config b/build/config/ws73_default.config
index b47819c..8bc0b3f 100644
--- a/build/config/ws73_default.config
+++ b/build/config/ws73_default.config
@@ -10,17 +10,17 @@
 #
 WSCFG_USING_GCC=y
 # WSCFG_USING_LLVM_CLANG is not set
-WSCFG_CROSS_COMPILE=""
-WSCFG_KERNEL_DIR=""
+WSCFG_CROSS_COMPILE="/home/mao/work/rk3588/orangepi-build/toolchains/gcc-arm-11.2-2022.02-x86_64-aarch64-none-linux-gnu/bin/aarch64-none-linux-gnu-"
+WSCFG_KERNEL_DIR="/home/mao/work/rk3588/orangepi-build/kernel/orange-pi-5.10-rk35xx"
 WSCFG_EXTRA_CFLAGS=""
-WSCFG_EXTRA_PARAMS=""
+WSCFG_EXTRA_PARAMS="-Wno-error"
 WSCFG_ARCH_ARM=y
 # WSCFG_ARCH_CUSTOM is not set
-WSCFG_ARCH_NAME="arm"
+WSCFG_ARCH_NAME="arm64"
 BOARD_ASIC=y
 BOARD_ASIC_WIFI=y
-WSCFG_BUS_SDIO=y
-# WSCFG_BUS_USB is not set
+# WSCFG_BUS_SDIO is not set
+WSCFG_BUS_USB=y
 # WSCFG_BUS_UART is not set
 WSCFG_LINUX=y
 _PRE_OS_VERSION_LINUX=0
@@ -137,7 +137,7 @@ CONFIG_HCC_SDIO_SUPPORT_SCATTER=y
 BT_EM_BUFFER_CALI_SUPPORT=y
 # _PRE_PLAT_SLP_UART_FORWARD is not set
 CONFIG_DFX_SUPPORT_SYSFS=y
-CONFIG_SDIO_RESCAN=y
+# CONFIG_SDIO_RESCAN is not set

 #
 # Configure the loading path of files such as firmware

```