IDO-EVB5301-V1 Linux开发手册

1 上手指南

1.介绍

2 Linux开发

2.1.获取 SDK

2.2.校验md5值

2.3.解压

2.4.SDK目录介绍

2.5.配置文件介绍

2.6 分区说明

2.7 ddr配置说明

3.编译

3.1.选择项目

3.2.自动编译

3.3.分步编译

3.3.1.编译uboot

3.3.2.编译kernel

3.3.3.编译buildroot系统

3.3.4.固件打包

4 烧录说明

4.1.整包烧录

4.1.1.运行全志烧录工具

4.1.2.点击一键刷机

4.1.3.选择固件

4.1.4.点击升级

4.2.分包烧录

4.2.1.选择整包固件

4.2.2.烧录

5.驱动开发

5.1.CAN
5.1.1.CAN简介
5.1.2.DTS 节点配置
配置CAN1
双CAN配置
通信测试
1.无法正常收发
2.CAN时钟频率配置
5.2.Ethernet
公共配置
板级的配置
5.3.LCD
dts修改:
5.4.RTC
5.5.UART
DTS配置
收发测试
6.开发案例
6.1.Qt5使用
6.2.MQTT使用
6.3.GDB使用
6.4. LVGL使用

Industio
触 觉 智 能

IDO–EVB5301–V1

Linux开发手册

深圳触觉智能科技有限公司

www.industio.cn

文档修订历史

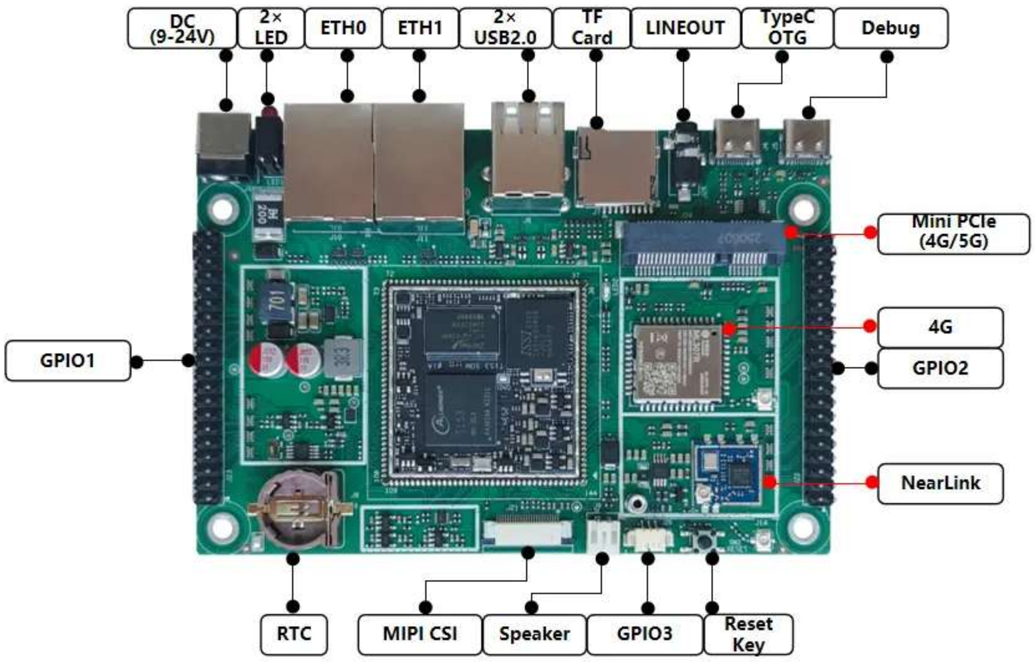
版本	PCBA版本	修订内容	修订	审核	日期
V1.0	V1A	创建文档	LJH	IDO	2025/10/21

1 上手指南

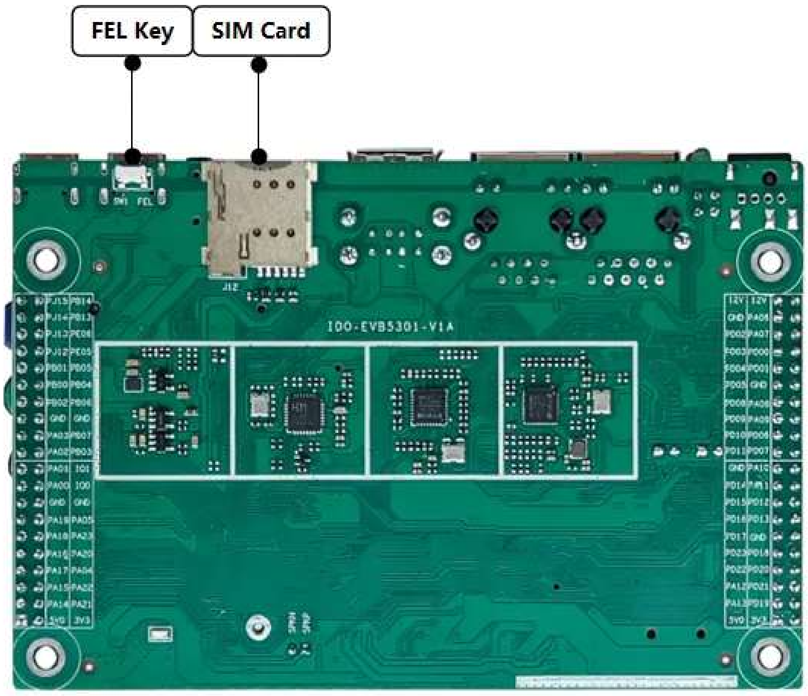
1.介绍

IDO-EVB5301-V1采用 Allwinner T153（四核 Cortex-A7, 主频最高1.6GHz）设计的评估开发板，支持DDR3/DDR3L/DDR4内存，T153可提供强大的计算性能和快速响应，完美胜任高要求的自动化任务。还配备一路千兆以太网和两路百兆以太网接口、两个CAN_FD接口和localbus，支持高吞吐量网络连接，

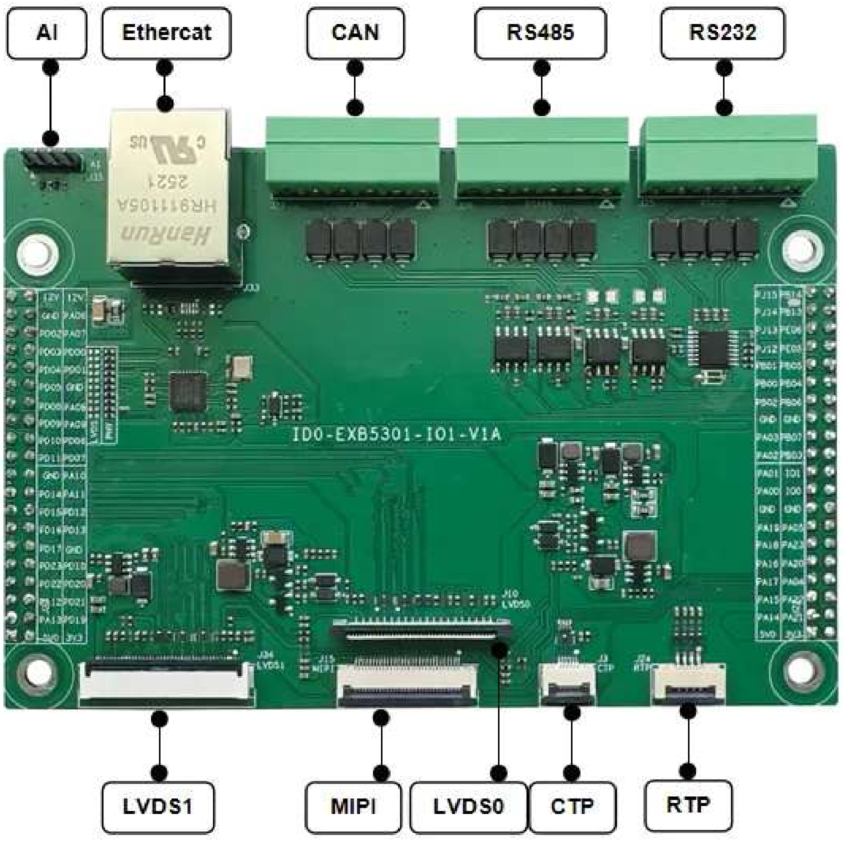
满足复杂数据驱动型应用需求。集成的图像信号处理器和显示引擎可为精密制造流程管理提供清晰的实时视觉反馈。适用于可编程逻辑控制器(PLC)、人机界面(HMI)机器人等场景。



IDO–EVB5301–V1正面



IDO–EVB5301–V1背面



IDO–EVB5301–V1 拓展板

2 Linux开发

2.1.获取 SDK

下载路径： `百度网盘/EVB5301/EVB5301-Linux/4.软件资料/SDK-V1.0（或SDK-V2.0）/系统SDK`

2.2.校验md5值

```bash
$ cat ./ido_evb5301_v0.95_a* > ido_evb5301_v0.95.tar.gz
$ md5sum ido_evb5301_v0.95.tar.gz
```

2.3.解压

首先准备一个空文件夹用于存放 SDK，建议在 home 目录下，本文以~/evb5301为例

Bash

```bash
$ mkdir ~/evb5301
$ cd ~/evb5301
$ tar -xvf ido_evb5301_v0.95.tar.gz -C ./
```

2.4.SDK目录介绍

```bash
/
├── brandy //uboot和boot0代码
│ ├── brandy-1.0
│ └── brandy-2.0
├── bsp
│ ├── config //soc芯片平台公共配置
│ ├── drivers //驱动接口相关代码
│ ├── modules //nand和gpu驱动目录
│ └── ramfs //mini-sys文件系统
├── build //编译打包脚本
│ ├── bin //制作文件系统工具
│ ├── createkeys //创建安全方案秘钥工具
│ ├── envsetup.sh //配置环境变量
│ ├── Makefile
│ ├── mkcmd.sh //主要编译脚本
│ ├── mkcommon.sh //编译脚本
│ ├── mksetup.sh
│ ├── pack //打包脚本
│ ├── toolchain //编译工具链
│ └── top_build.sh
├── build.sh -> build/top_build.sh //顶级编译脚本，链接到build/top_build.sh
├── device //方案配置文档
│ ├── config
│ | |chips //板级配置
│ | | ├── a40i //A40I配置
│ | | | ├── bin //uboot和烧写程序
│ | | | ├── boot-resource //启动资源文件，如bootlogo
│ | | | ├── config //方案板配置
│ | | | | ├── default //默认配置和通用配置
│ | | | | ├── p3 //a40i p3方案板配置
│ | | | | | ├── buildroot //linux sdk配置
│ | | | | | | ├── swupdate //linux OTA升级配置文件
│ ├── product -> ./config/chips/a40i
│ └── target //方案配置目录
├── kernel //各个版本的内核文件
│ ├── linux-4.9
│ ├── linux-5.4
│ ├── linux-5.10
├── out
│ ├── gcc-linaro-* //kernel交叉编译工具链
│ |── pack_out //打包输出目录
│ ├── a40i //a40i输出目录
│ | |── p3 //a40i p3输出目录
│ | | |── bsp //bsp输出目录
│ ├── a40i_linux_p3_uart0.img //a40i生成固件
```

```
├── test
│ ├── dragonboard //dragonboard测试系统
│ └── SATA //sata测试系统
└── tools //pc工具
| ├── build
| ├── codecheck //代码检查工具
| ├── pack
| └── tools_win //windows软件工具
```

注意:

1. SDK 采用交叉编译，所以要在 X86_64 电脑上使用 SDK，不要将 SDK 下载到板子上。
2. 编译环境请使用 Ubuntu22.04（真机或 虚拟机），如果使用其他版本可能导致编译出错。
3. 不要在虚拟机共享文件夹以及非英文目录存放、解压SDK。
4. 获取、编译 SDK 请全程使用普通用户，不允许也不需要使用 root 权限（除非需要 apt 安装软件）

2.5.配置文件介绍

当 `./build.sh config` 完成之后，编译系统会生成一个方案的编译规则，其保存在SDK根目录中的 `.buildconfig` 文件里面。下面是一个规则的说明。

```bash
LICHEE_PLATFORM //编译平台
LICHEE_LINUX_DEV //编译的方案
LICHEE_IC // 编译的IC
LICHEE_BOARD //编译的板级配置
LICHEE_FLASH //编译的flash配置
LICHEE_CHIP //编译的芯片代号名称
LICHEE_KERN_VER //编译的内核版本
LICHEE_KERN_DEFCONF //编译内核的默认配置
LICHEE_BUILDING_SYSTEM //编译的构造系统
LICHEE_BR_VER //buildroot的版本
LICHEE_BR_DEFCONF //buildroot的默认配置
LICHEE_BR_RAMFS_CONF //buildroot的ramfs默认配置
LICHEE_BRANDY_VER //uboot的版本
LICHEE_BRANDY_DEFCONF //brandy的默认配置
LICHEE_CROSS_COMPILER //编译使用的交叉编译链
LICHEE_CHIP_CONFIG_DIR //编译系统的IC配置目录
LICHEE_OUT_DIR //编译的输出目录
```

上面只列出了部分重要的配置，详细的配置可以查看该文件。
而某个方案的大部分编译规则都是由 BoardConfig.mk，其存放在每个方案的板级配置目录中，同一个板型的不同系统拥有不用的 BoardConfig.mk 配置，例如 T153 bsp 验证系统和 Linux SDK 系统拥有各自的 BoardConfig.mk 配置。

2.6 分区说明

`<SDK_TOP_PATH>/device/config/chips/t153/configs/{board}/buildroot/sys_partition.fex` 文件中包含了固件的分区信息：

```bash
;----------------------------------------------------------------------------------------------------
; 说明： 脚本中的字符串区分大小写，用户可以修改"="后面的数值，但是不要修改前面的字符串
;----------------------------------------------------------------------------------------------------


;----------------------------------------------------------------------------------------------------
;                                   固件下载参数配置
;----------------------------------------------------------------------------------------------------
;****************************************************************************************************
;    mbr的大小, 以Kbyte为单位
;****************************************************************************************************
[mbr]
size = 16384

;****************************************************************************************************
;                                              分区配置
;
;
;  partition 定义范例:
;    [partition]                ;  //表示是一个分区
;    name        = USERFS2      ; //分区名称
;    size        = 16384        ; //分区大小 单位: 扇区.分区表示个数最多2^31 * 512 = 2T
;    downloadfile = "123.fex"   ; //下载文件的路径和名称，可以使用相对路径，相对是指相对于image.cfg文件所在分区。也可以使用绝对路径
;    keydata     = 1            ; //私有数据分区，重新量产数据将不丢失
;    encrypt     = 1            ; //采用加密方式烧录，将提供数据加密，但损失烧录速度
;    user_type   = ?            ; //私有用法
;    verify      = 1            ; //要求量产完成后校验是否正确
;
; 注：1、name唯一, 不允许同名
;      2、name最大12个字符
;      3、size = 0, 将创建一个无大小的空分区
;      4、为了安全和效率考虑，分区大小最好保证为16M字节的整数倍
;****************************************************************************************************
[partition_start]
```

```

[partition]
    name         = boot-resource
    size         = 65536
    downloadfile = "boot-resource.fex"
    user_type    = 0x8000


[partition]
    name         = env
    size         = 1024
    downloadfile = "env.fex"
  user_type    = 0x8000

[partition]
    name         = env-redund
    size         = 1024
    downloadfile = "env.fex"
  user_type    = 0x8000

[partition]
    name         = boot
    size         = 65536
    downloadfile = "boot.fex"
  user_type    = 0x8000

[partition]
    name         = dtbo
    size         = 2048
    downloadfile = "dtbo.fex"
    user_type    = 0x8000

[partition]
    name         = dtbo-r
    size         = 2048
    downloadfile = "dtbo.fex"
    user_type    = 0x8000

[partition]
    name         = rootfs
    size         = 1048576
    downloadfile = "rootfs.fex"
  user_type    = 0x8000

[partition]
    name         = private
    size         = 248
    ro           = 0
```

```
        user_type     = 0x8000

```

2.7 ddr配置说明

开发板默认适配了以下两款ddr芯片

`A3T4GF40BBF-HPI,DDR3,容量512M,x16,1866Mbps,96BGA,工作温度-40℃~+95℃,Zentel(力积电子)`

`IS43TR16128DL-125KBLI,DDR3L,容量256M,128Mbx16,1600Mbps,96BGA,工作温度-40℃~+95℃,ISSI(北京君正)`

两款ddr芯片的初始化配置并不是统一的，ddr初始化配置位于以下两个配置文件中 `device/config/chips/t153/configs/bga_demo/sys_config.fex` ， `device/config/chips/t153/configs/bga_demo_nand/sys_config.fex`

编译固件默认会识别 `[dram_para]` 下的ddr配置。

力积DDR

Shell

```
[dram_para]
dram_clk          = 900
dram_type         = 3
dram_dx_odt       = 0x00000404
dram_dx_dri       = 0x00000808
dram_ca_dri       = 0x0c0c0c
dram_para0        = 0x00009898
dram_para1        = 0x30fa
dram_para2        = 0x0001
dram_mr0          = 0x840
dram_mr1          = 0x42
dram_mr2          = 0x8
dram_mr3          = 0x0
dram_mr4          = 0x0
dram_mr5          = 0x0
dram_mr6          = 0x0
dram_mr11         = 0x0
dram_mr12         = 0x0
dram_mr13         = 0x0
dram_mr14         = 0x0
dram_mr16         = 0x0
dram_mr17         = 0x0
dram_mr22         = 0x0
dram_tpr0         = 0x0
dram_tpr1         = 0x0
dram_tpr2         = 0x0
dram_tpr3         = 0x0
dram_tpr6         = 0x40
dram_tpr10        = 0x00001e00
dram_tpr11        = 0x00009494
dram_tpr12        = 0x13131313
dram_tpr13        = 0x08007071
dram_tpr14        = 0x810700f5
```

君正ddr

Shell

```shell
[dram_para]
dram_clk          = 800
dram_type         = 3
dram_dx_odt       = 0x00000404
dram_dx_dri       = 0x00000808
dram_ca_dri       = 0x0c0c0c
dram_para0        = 0x00009898
dram_para1        = 0x30ea
dram_para2        = 0x0001
dram_mr0          = 0x840
dram_mr1          = 0x42
dram_mr2          = 0x8
dram_mr3          = 0x0
dram_mr4          = 0x0
dram_mr5          = 0x0
dram_mr6          = 0x0
dram_mr11         = 0x0
dram_mr12         = 0x0
dram_mr13         = 0x0
dram_mr14         = 0x0
dram_mr16         = 0x0
dram_mr17         = 0x0
dram_mr22         = 0x0
dram_tpr0         = 0x0
dram_tpr1         = 0x0
dram_tpr2         = 0x0
dram_tpr3         = 0x0
dram_tpr6         = 0x40
dram_tpr10        = 0x00001e00
dram_tpr11        = 0x00009293
dram_tpr12        = 0x0000191B
dram_tpr13        = 0x08007071
dram_tpr14        = 0x810700f5
```

3.编译

3.1.选择项目

```bash
./build.sh config
10-24 16:25:17.805  158713 D mkcommon  : ========ACTION List: mk_config ;=
=======
10-24 16:25:17.806  158713 D mkcommon  : options :
All available platform:
   0. android
   1. linux
Choice [linux]: 1 //选择编译linux固件，按实际选择
All available linux_dev:
   0. bsp
   1. buildroot
   2. ubuntu
Choice [buildroot]: 1 //选择bsp方案，按实际选择
All available ic:
   0. t153
Choice [t153]: 0
All available board:
   0. bga_demo
   1. bga_demo_amp_nand
   2. bga_demo_nand
   3. bga_demo_nor
   4. demo
   5. demo2_nand
   6. demo_amp_nand
   7. demo_nand
   8. demo_nor
   9. demo_qa
Choice [bga_demo]: 0  //选择bsp方案，目前支支持0和2;0: emmc版本;2: nand_flash版
本
All available flash:
   0. default
   1. nor
Choice [default]: 0 //选择存储介质，按实际选择
All available kern_name:
   0. linux-5.10-origin
   1. linux-5.10-rt
   2. linux-5.10-xenomai
Choice [linux-5.10-origin]: 0 //选择linux内核版本，按实际选择
```

选择对应的数字即可。

3.2.自动编译

```bash
$ ./build.sh
```

3.3.分步编译

3.3.1.编译uboot

```bash
./build.sh uboot
```

3.3.2.编译kernel

```bash
./build.sh kernel
```

3.3.3.编译buildroot系统

```bash
./build.sh buildroot
```

3.3.4.固件打包

```bash
./build.sh pack
```

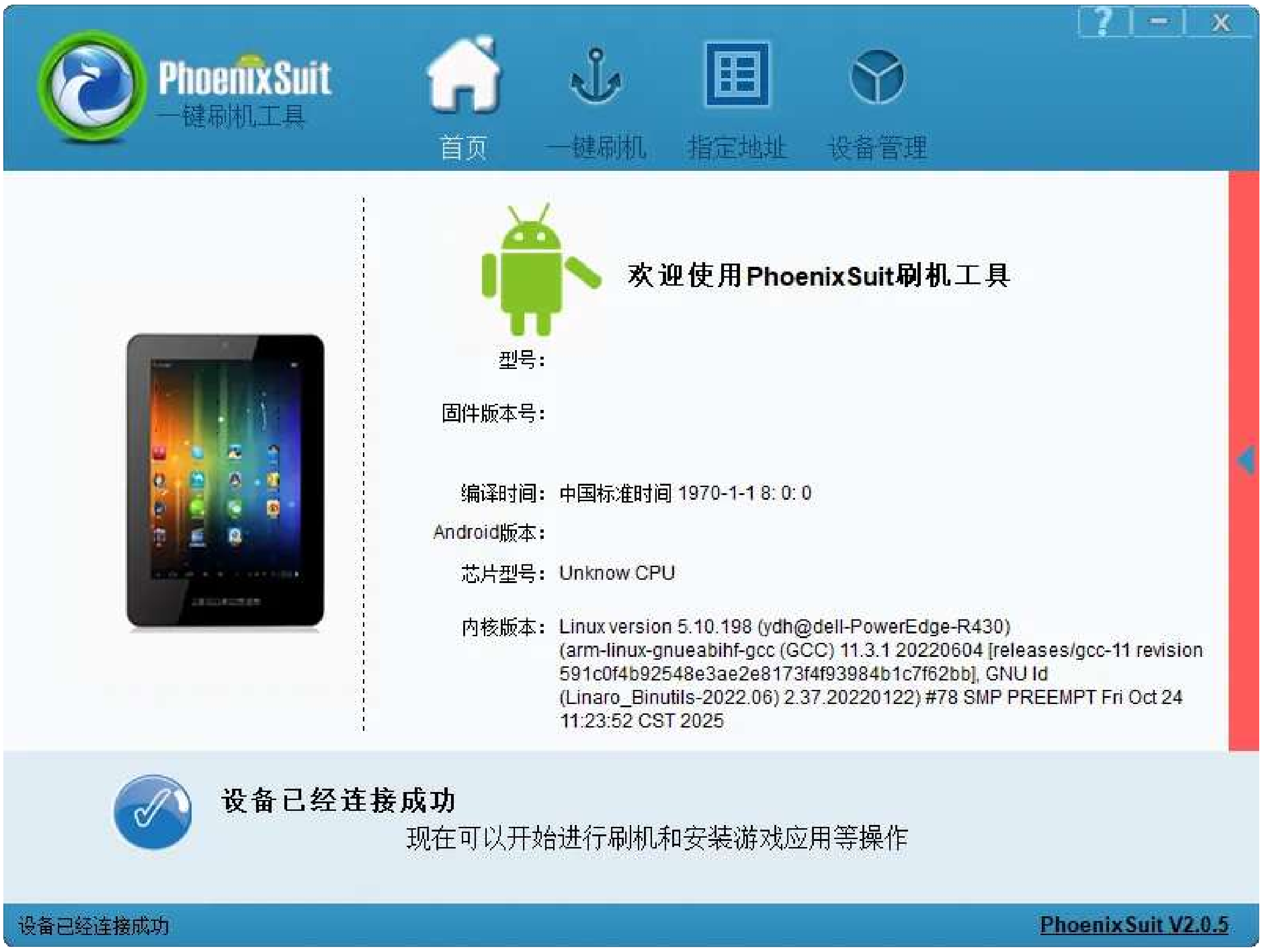
编译生成的固件在SDK中 /out 目录下 t53_linux_bga_demo_uart0.img

4 烧录说明

注意：板子需要进入到烧录模式，具体方法可以参考：《IDO–EVB5301–V1开发板固件烧录手册》

4.1.整包烧录

4.1.1.运行全志烧录工具



欢迎使用PhoenixSuit刷机工具

型号：

固件版本号：

编译时间：中国标准时间 1970-1-1 8: 0: 0

Android版本：

芯片型号：Unknow CPU

内核版本：Linux version 5.10.198 (ydh@dell-PowerEdge-R430) (arm-linux-gnueabihf-gcc (GCC) 11.3.1 20220604 [releases/gcc-11 revision 591c0f4b92548e3ae2e8173f4f93984b1c7f62bb], GNU ld (Linaro_Binutils-2022.06) 2.37.20220122) #78 SMP PREEMPT Fri Oct 24 11:23:52 CST 2025

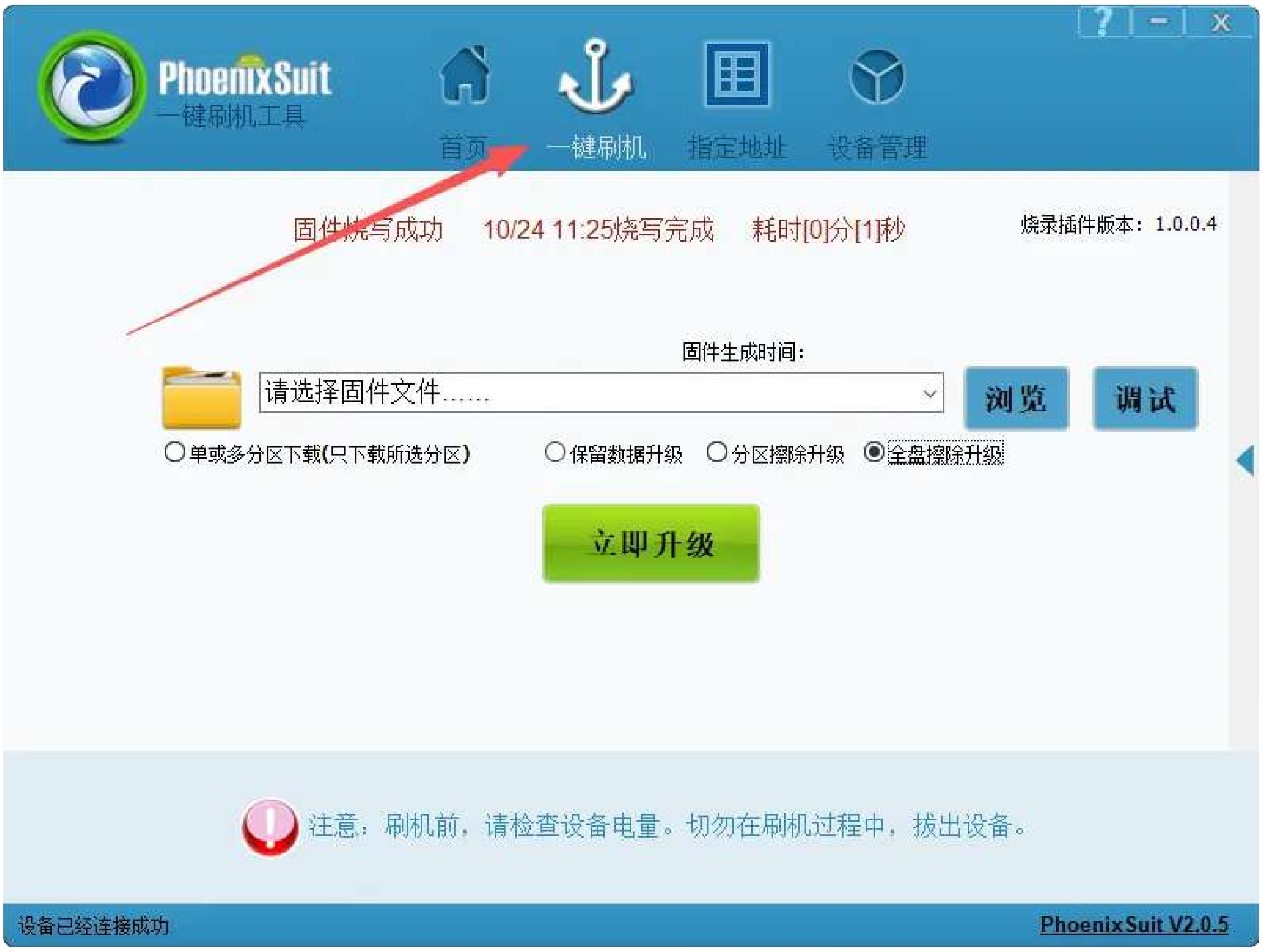
设备已经连接成功

现在可以开始进行刷机和安装游戏应用等操作

设备已经连接成功

PhoenixSuit V2.0.5

4.1.2.点击一键刷机



固件烧写成功　10/24 11:25烧写完成　耗时[0]分[1]秒

烧录插件版本：1.0.0.4

固件生成时间：

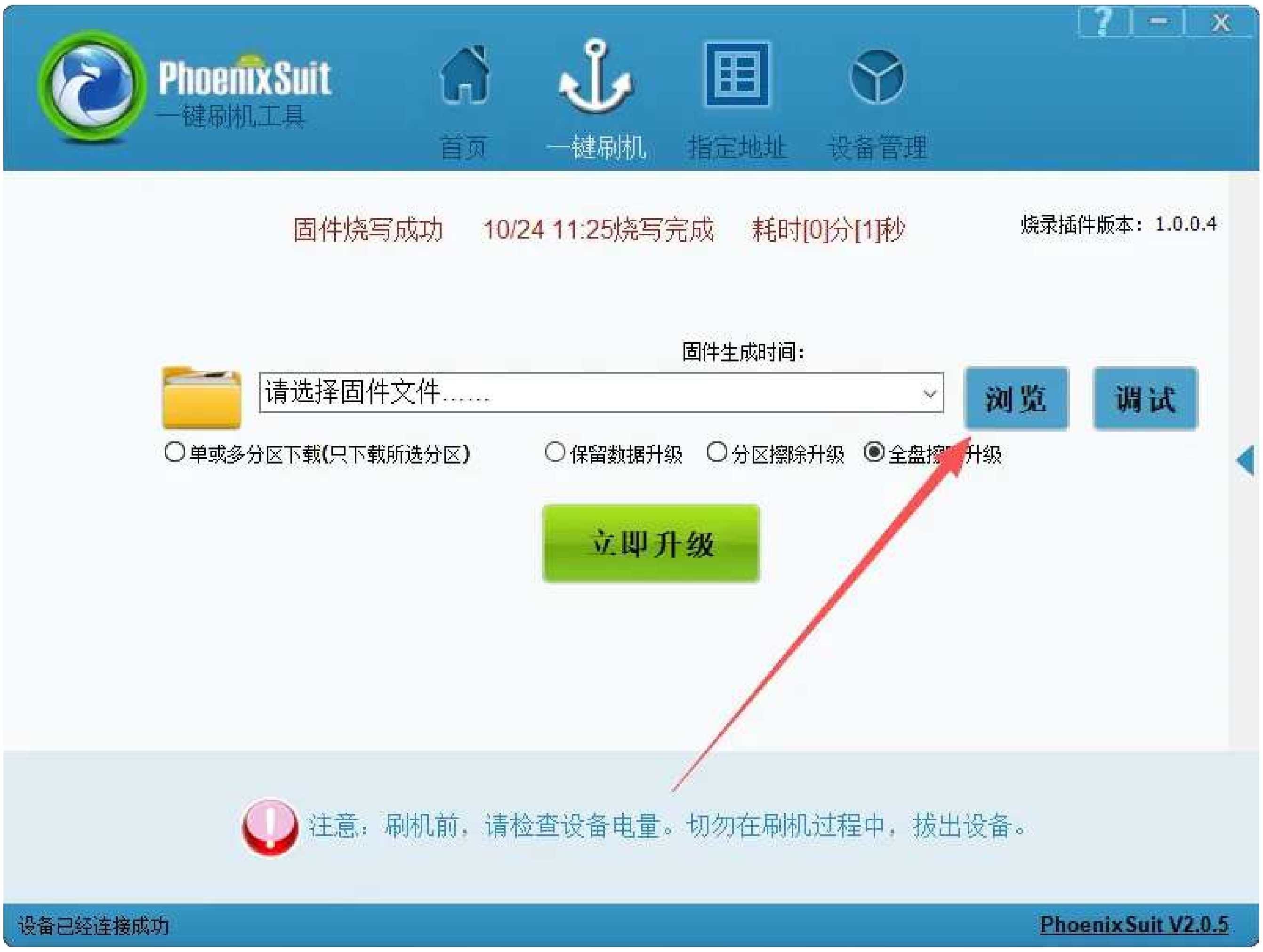
请选择固件文件......

○单或多分区下载(只下载所选分区)　○保留数据升级　○分区擦除升级　◉全盘擦除升级

注意：刷机前，请检查设备电量。切勿在刷机过程中，拔出设备。

设备已经连接成功　PhoenixSuit V2.0.5

4.1.3.选择固件



固件烧写成功 10/24 11:25烧写完成 耗时[0]分[1]秒

烧录插件版本：1.0.0.4

固件生成时间：

请选择固件文件......

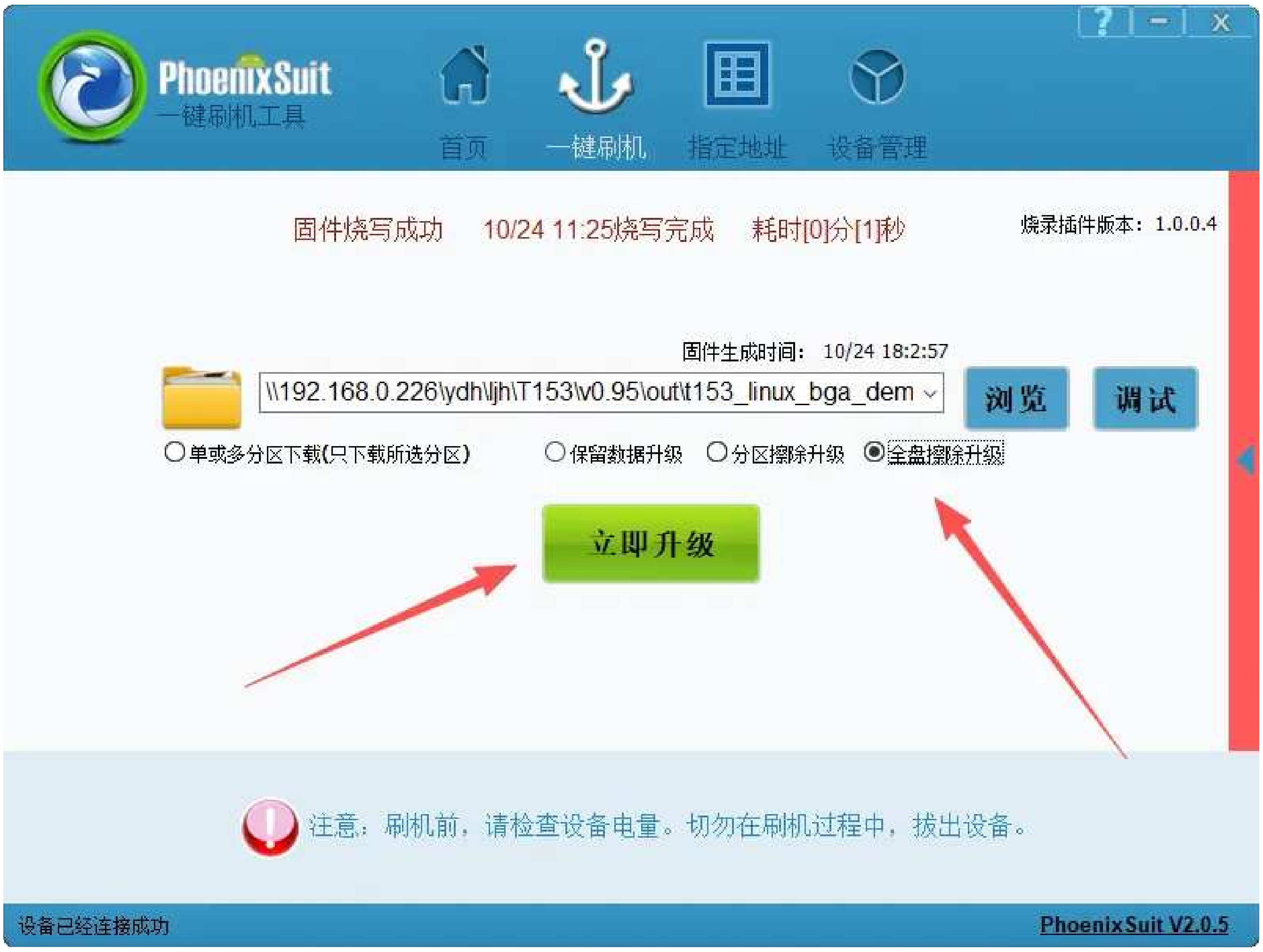
○单或多分区下载(只下载所选分区) ○保留数据升级 ○分区擦除升级 [✓ ●全盘擦除升级]

注意：刷机前，请检查设备电量。切勿在刷机过程中，拔出设备。

设备已经连接成功 PhoenixSuit V2.0.5

选择： `out/t153_linux_{board}_uart0.img`

4.1.4.点击升级



固件烧写成功　10/24 11:25烧写完成　耗时[0]分[1]秒

烧录插件版本：1.0.0.4

固件生成时间：10/24 18:2:57

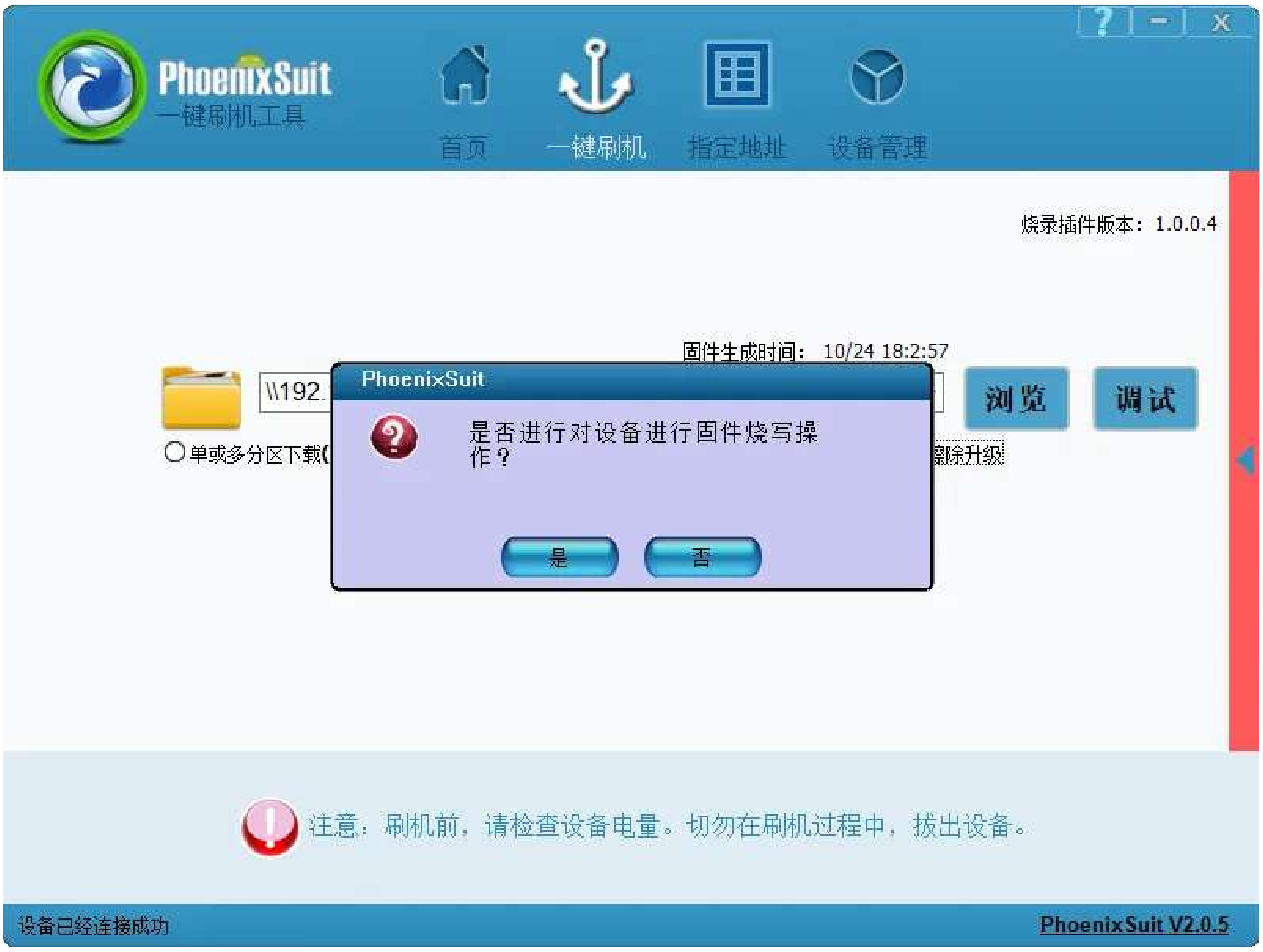
\\192.168.0.226\ydh\ljh\T153\v0.95\out\t153_linux_bga_dem

○单或多分区下载(只下载所选分区)　○保留数据升级　○分区擦除升级　[✓ 全盘擦除升级]

注意：刷机前，请检查设备电量。切勿在刷机过程中，拔出设备。

设备已经连接成功　　PhoenixSuit V2.0.5

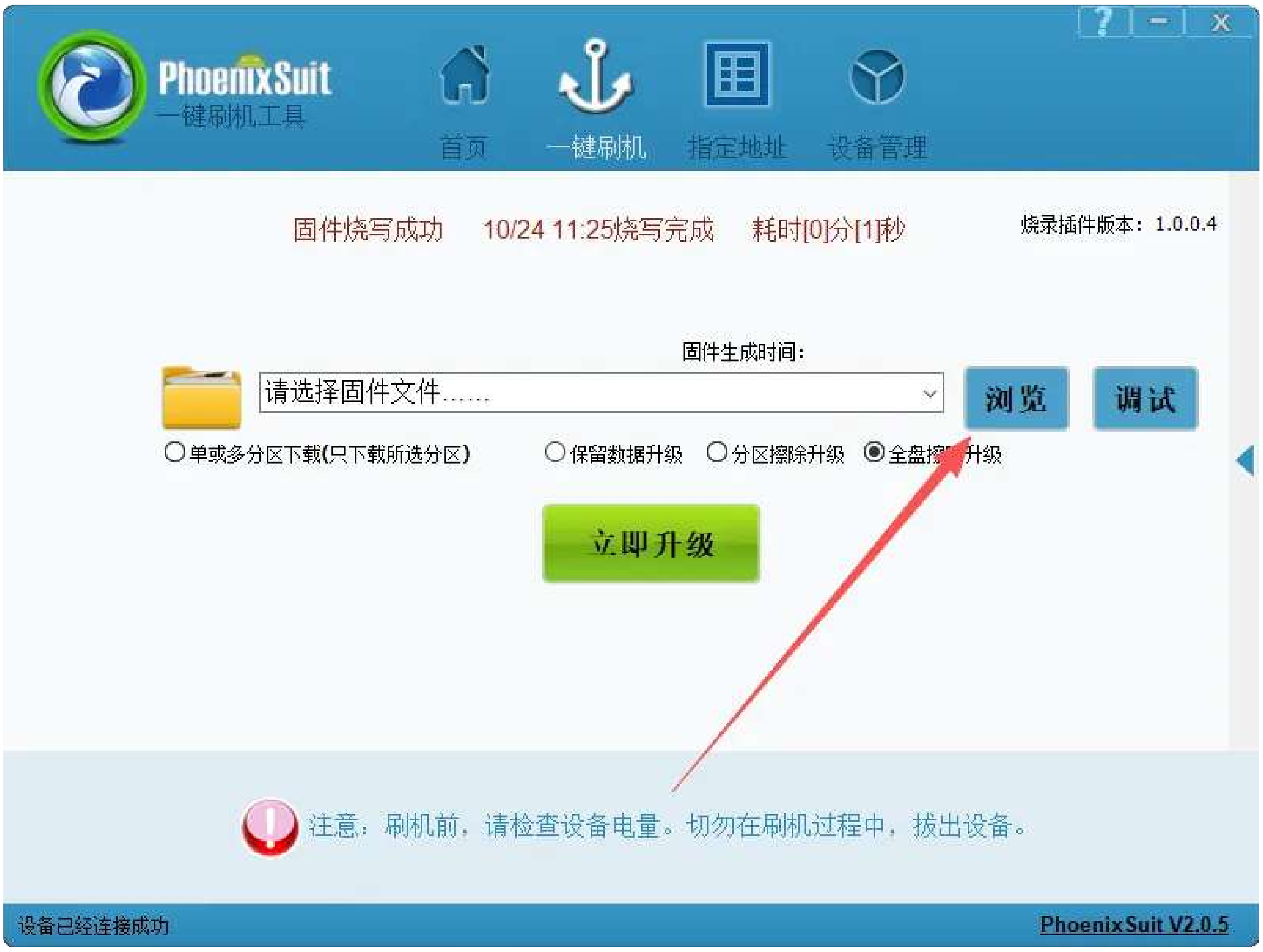
勾选全盘擦除升级后，点击立即升级。



工具提示是否进行对设备镜像烧写工作，点击是，自动开始烧录完整固件。

4.2.分包烧录

4.2.1.选择整包固件



固件烧写成功　10/24 11:25烧写完成　耗时[0]分[1]秒

烧录插件版本：1.0.0.4

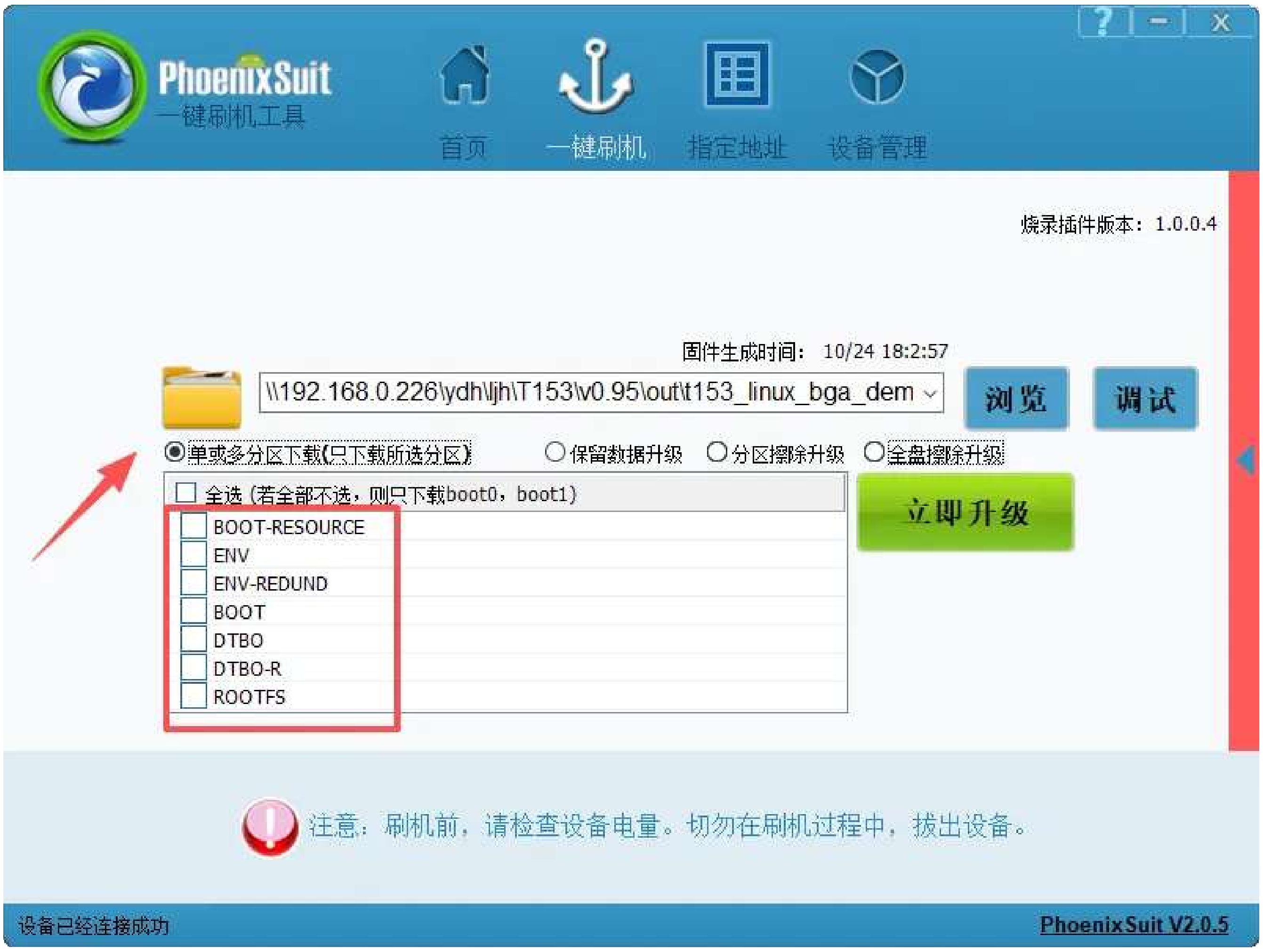
固件生成时间：

请选择固件文件......

○单或多分区下载(只下载所选分区)　○保留数据升级　○分区擦除升级　● 全盘擦除升级

注意：刷机前，请检查设备电量。切勿在刷机过程中，拔出设备。

设备已经连接成功　PhoenixSuit V2.0.5


烧录插件版本：1.0.0.4

固件生成时间：10/24 18:2:57

\\192.168.0.226\ydh\ljh\T153\v0.95\out\t153_linux_bga_dem

[✓ 单或多分区下载(只下载所选分区)] 保留数据升级 分区擦除升级 全盘擦除升级

☐ 全选（若全部不选，则只下载boot0，boot1）
☐ BOOT-RESOURCE
☐ ENV
☐ ENV-REDUND
☐ BOOT
☐ DTBO
☐ DTBO-R
☐ ROOTFS

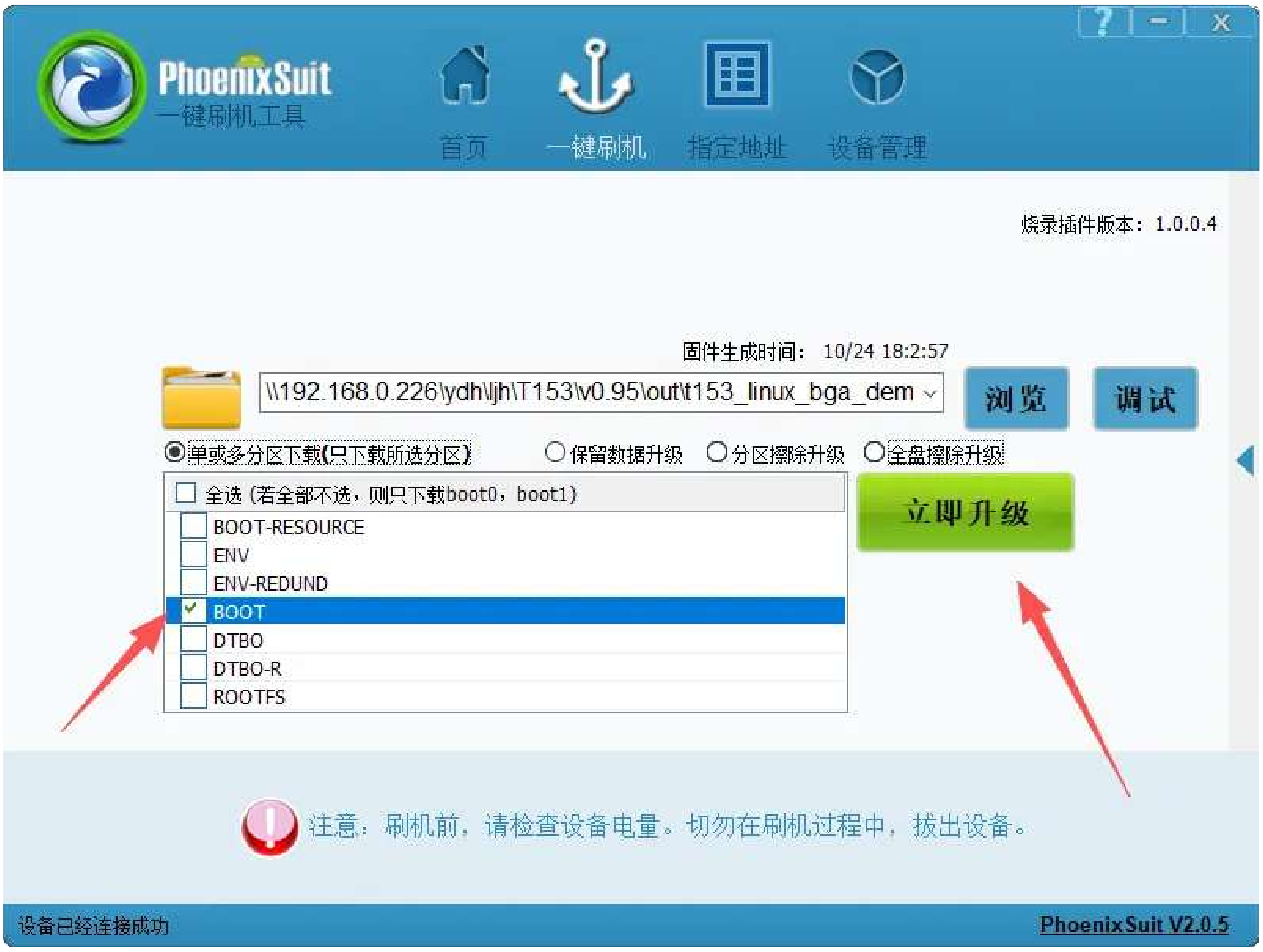
注意：刷机前，请检查设备电量。切勿在刷机过程中，拔出设备。

设备已经连接成功

选择单或多分区下载，勾选需要单独烧录的分区，并点击立刻升级即可。

注：这里的分区由整包固件的分区表决定，烧录前需保证分区表一致。

4.2.2.烧录



烧录插件版本：1.0.0.4

固件生成时间：10/24 18:2:57

\\192.168.0.226\ydh\ljh\T153\v0.95\out\t153_linux_bga_dem

● 单或多分区下载(只下载所选分区)　○ 保留数据升级　○ 分区擦除升级　○ 全盘擦除升级

- ☐ 全选（若全部不选，则只下载boot0，boot1）
- ☐ BOOT-RESOURCE
- ☐ ENV
- ☐ ENV-REDUND
- ☑ BOOT
- ☐ DTBO
- ☐ DTBO-R
- ☐ ROOTFS

注意：刷机前，请检查设备电量。切勿在刷机过程中，拔出设备。

设备已经连接成功　　PhoenixSuit V2.0.5

以单独烧录boot分区为例，单独勾选 `boot` 选项后，手动选择立即升级即可。

5.驱动开发

5.1.CAN

5.1.1.CAN简介

CAN(Controller Area Network)总线，即控制器局域网总线，是一种有效支持分布式控制或实时控制的串行通信网络。CAN总线是一种在汽车上广泛采用的总线协议，被设计作为汽车环境中的微控制器通讯。

5.1.2.DTS 节点配置

```
<SDK_TOP_PATH>/device/config/chips/t153/configs/{board}/board.dts
```

```bash
	can0 {
		compatible = "allwinner,sun8i-t153-can";
		reg = <0x0 0x0453C800 0x0 0x400>,
			<0x0 0x04538000 0x0 0x4400>,
			<0x0 0x04530000 0x0 0x1000>,
			<0x0 0x04534000 0x0 0x1000>;
		reg-names = "can", "message_ram", "can_top", "can_dma";
		interrupts = <GIC_SPI 36 IRQ_TYPE_LEVEL_HIGH>, <GIC_SPI 35 IRQ_TYPE_LEVEL_HIGH>, <GIC_SPI 37 IRQ_TYPE_LEVEL_HIGH>;
		interrupt-names = "int0", "int_top", "int1";
		clocks = <&ccu CLK_CAN0>, <&ccu CLK_BUS_CAN0>, <&ccu CLK_MBUS_CAN_GATE>;
		clock-names = "can_clk", "can_bus", "can_mbus";
		resets = <&ccu RST_BUS_CAN0>, <&ccu RST_BUS_CAN_SYS>;
		reset-names = "can_rst", "can_sys_rst";
		clock-frequency = <40000000>;
		allwinner,ram-cfg = <0x0 0 0 64 64 64 32 32>;
		pinctrl-0 = <&can0_pins_active>;
		pinctrl-1 = <&can0_pins_sleep>;
		pinctrl-names = "default", "sleep";
		status = "disabled";
	};

	can1 {
		compatible = "allwinner,sun8i-t153-can";
		reg = <0x0 0x04541800 0x0 0x400>,
				<0x0 0x0453D000 0x0 0x4400>,
				<0x0 0x04531000 0x0 0x1000>,
				<0x0 0x04535000 0x0 0x1000>;
		reg-names = "can", "message_ram", "can_top", "can_dma";
		interrupts = <GIC_SPI 144 IRQ_TYPE_LEVEL_HIGH>, <GIC_SPI 146 IRQ_TYPE_LEVEL_HIGH>, <GIC_SPI 145 IRQ_TYPE_LEVEL_HIGH>;
		interrupt-names = "int0", "int_top", "int1";
		clocks = <&ccu CLK_CAN1>, <&ccu CLK_BUS_CAN1>, <&ccu CLK_MBUS_CAN_GATE>;
		clock-names = "can_clk", "can_bus", "can_mbus";
		resets = <&ccu RST_BUS_CAN1>, <&ccu RST_BUS_CAN_SYS>;
		reset-names = "can_rst", "can_sys_rst";
		clock-frequency = <40000000>;
		allwinner,ram-cfg = <0x0 0 0 64 64 64 32 32>;
		pinctrl-0 = <&can1_pins_active>;
		pinctrl-1 = <&can1_pins_sleep>;
		pinctrl-names = "default", "sleep";
		status = "disabled";
	};
```

```
&pio {


};
```

配置CAN1

```
&can0 {
    status = "disabled";
};

&can1 {
    status = "okay";
};
```

双CAN配置

板级配置：ido_evb3506_v1a-emmc.dtsi

```
&can0 {
    status = "okay";
};

&can1 {
    status = "okay";
};
```

通信测试

```bash
#检查can设备
$ ip link show

#在收发端关闭can0设备
$ ip link set can0 down

#设置仲裁段1M波特率，数据段1M波特率
$ ip link set can0 type can bitrate 1000000 dbitrate 1000000 fd on
[   63.834376] rk3576_canfd ff330000.can can0: bitrate error 0.3%
[   63.834512] rk3576_canfd ff330000.can can0: bitrate error 0.3%

#在收发端打开can0设备
$ ip link set can0 up

#在接收端执行candump,阻塞等待报文
$ candump can0

#在发送端执行cansend，发送报文
$ cansend can0 123#1122334455667788

#检查是否启用成功
$ ip link show can0
```

总结调试过程中遇到的几个问题及解决方法：

1.无法正常收发

检查总线 CAN_H 和 CAN_L， 杜邦线是否松动或者接反。

2.CAN时钟频率配置

如果CAN的比特率低于等于3M建议修改CAN时钟到100M,信号更稳定。高于3M比特率的, 时钟设置200M就可以。

CAN时钟频率修改方法参考如下：

```bash
&can0 {
……
-         clock-frequency = <300000000>;
+         clock-frequency = <200000000>;
……
 };

&can1 {
……
-         clock-frequency = <300000000>;
+         clock-frequency = <200000000>;
……
 };
```

在某些时钟频率下，CAN的bitrate无法获得准确的速率，可以自行调整clock-frequency去解决。

查看是否得到所需的bitrare：

```bash
ip -d link show can0
```

5.2.Ethernet

公共配置

`bsp/configs/{kernel}/sun8iw22p1.dtsi`

Bash

```bash
    gmac0: ethernet@4500000 {
      compatible = "allwinner,sunxi-gmac-211";
      reg = <0x0 0x04500000 0x0 0x8000>, <0x0 0x04508000 0x0 0x1000>;
      interrupts = <GIC_SPI 46 IRQ_TYPE_LEVEL_HIGH>;
      interrupt-names = "macirq";
      clocks = <&ccu CLK_BUS_GMAC0_AHB>, <&ccu CLK_MBUS_BUS_GMAC0>, <&ccu CLK_GMAC0_PHY>, <&ccu CLK_GMAC0_PTP_REF>,
          <&ccu CLK_BUS_GMAC0_AHB_SW_CFG>, <&ccu CLK_MBUS_BUS_GMAC0_SW_CFG>;
      clock-names = "stmmaceth", "pclk", "phy", "ptp_ref",
            "ahb_sw_cfg", "mhub_sw_cfg";
      assigned-clocks = <&ccu CLK_GMAC0_PHY>;
      assigned-clock-rates = <25000000>;
      resets = <&ccu RST_BUS_GMAC0_AXI>, <&ccu RST_BUS_GMAC0_AHB>;
      reset-names = "stmmaceth", "ahb";
      phy-mode = "rgmii";
      phy-handle = <&gmac0_phy0>;
      status = "disabled";

      aw,rgmii-clk-ext;
      snps,fixed-burst;
      snps,en-tx-lpi-clockgating;
      snps,axi-config = <&gmac_stmmac_axi_setup>;
      snps,mtl-rx-config = <&gmac_mtl_rx_setup>;
      snps,mtl-tx-config = <&gmac_mtl_tx_setup>;

      mdio0: mdio0@0 {
        compatible = "snps,dwmac-mdio";
        #address-cells = <1>;
        #size-cells = <0>;
        gmac0_phy0: ethernet-phy@1 {
          compatible = "ethernet-phy-ieee802.3-c22";
          reg = <0x1>;
          max-speed = <1000>;  /* Max speed capability */
          reset-gpios = <&pio PA 14 GPIO_ACTIVE_LOW>;
          /* PHY datasheet rst time */
          reset-assert-us = <10000>;
          reset-deassert-us = <150000>;
        };
      };
    };

    gmac1: ethernet@4510000 {
      compatible = "allwinner,sunxi-gmac-211";
      reg = <0x0 0x04510000 0x0 0x8000>, <0x0 0x04518000 0x0 0x1000>;
```

```
        interrupts = <GIC_SPI 47 IRQ_TYPE_LEVEL_HIGH>;
        interrupt-names = "macirq";
        clocks = <&ccu CLK_BUS_GMAC1_AHB>, <&ccu CLK_MBUS_BUS_GMAC1>, <&cc
u CLK_GMAC1_PHY>, <&ccu CLK_GMAC1_PTP_REF>,
            <&ccu CLK_BUS_GMAC1_AHB_SW_CFG>, <&ccu CLK_MBUS_BUS_GMAC1_SW_CF
G>;
        clock-names = "stmmaceth", "pclk", "phy", "ptp_ref",
              "ahb_sw_cfg", "mhub_sw_cfg";
        assigned-clocks = <&ccu CLK_GMAC1_PHY>;
        assigned-clock-rates = <25000000>;
        resets = <&ccu RST_BUS_GMAC1_AXI>, <&ccu RST_BUS_GMAC1_AHB>;
        reset-names = "stmmaceth", "ahb";
        phy-mode = "rgmii";
        phy-handle = <&gmac1_phy0>;
        status = "disabled";

        aw,rgmii-clk-ext;
        snps,fixed-burst;
        snps,en-tx-lpi-clockgating;
        snps,axi-config = <&gmac_stmmac_axi_setup>;
        snps,mtl-rx-config = <&gmac_mtl_rx_setup>;
        snps,mtl-tx-config = <&gmac_mtl_tx_setup>;

        mdio1: mdio1@0 {
          compatible = "snps,dwmac-mdio";
          #address-cells = <1>;
          #size-cells = <0>;
          gmac1_phy0: ethernet-phy@1 {
            compatible = "ethernet-phy-ieee802.3-c22";
            reg = <0x1>;
            max-speed = <1000>;  /* Max speed capability */
            /* PHY datasheet rst time */
            reset-assert-us = <10000>;
            reset-deassert-us = <150000>;
          };
        };
      };

      gmac2: ethernet@4520000 {
        compatible = "allwinner,sunxi-gmac-211";
        reg = <0x0 0x04520000 0x0 0x8000>, <0x0 0x04528000 0x0 0x1000>;
        interrupts = <GIC_SPI 101 IRQ_TYPE_LEVEL_HIGH>;
        interrupt-names = "macirq";
        clocks = <&ccu CLK_BUS_GMAC2_AHB>, <&ccu CLK_MBUS_BUS_GMAC2>, <&cc
u CLK_GMAC2_PHY>, <&ccu CLK_GMAC2_PTP_REF>,
            <&ccu CLK_BUS_GMAC2_AHB_SW_CFG>, <&ccu CLK_MBUS_BUS_GMAC2_SW_CF
G>;
        clock-names = "stmmaceth", "pclk", "phy", "ptp_ref",
                "ahb_sw_cfg", "mhub_sw_cfg";
        assigned-clocks = <&ccu CLK_GMAC2_PHY>;
        assigned-clock-rates = <25000000>;
        resets = <&ccu RST_BUS_GMAC2_AXI>, <&ccu RST_BUS_GMAC2_AHB>;
        reset-names = "stmmaceth", "ahb";
        phy-mode = "rgmii";
        phy-handle = <&gmac2_phy0>;
        status = "disabled";

        aw,rgmii-clk-ext;
        snps,fixed-burst;
        snps,en-tx-lpi-clockgating;
        snps,axi-config = <&gmac_stmmac_axi_setup>;
        snps,mtl-rx-config = <&gmac_mtl_rx_setup>;
        snps,mtl-tx-config = <&gmac_mtl_tx_setup>;

        mdio2: mdio2@0 {
          compatible = "snps,dwmac-mdio";
          #address-cells = <1>;
          #size-cells = <0>;
          gmac2_phy0: ethernet-phy@1 {
            compatible = "ethernet-phy-ieee802.3-c22";
            reg = <0x1>;
            max-speed = <1000>;  /* Max speed capability */
            /* PHY datasheet rst time */
            reset-assert-us = <10000>;
            reset-deassert-us = <150000>;
          };
        };
      };
```

板级的配置

```bash
&gmac0_phy0 {
        compatible = "ethernet-phy-id001c.c916";
        reset-gpios = <&pio PJ 10 GPIO_ACTIVE_LOW>;
        /* For RTL8211F: PHY datasheet rst time */
        reset-assert-us = <10000>;
        reset-deassert-us = <30000>;
        reg = <0x1>;
        max-speed = <100>;
};

&gmac0 {
        phy-mode = "rmii";
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&gmac0_pins_default>;
        pinctrl-1 = <&gmac0_pins_sleep>;
        phy-handle = <&gmac0_phy0>;
        status = "okay";
};

&gmac1_phy0 {
        compatible = "ethernet-phy-id4f51.e91b";
        reg = <0x1>;
        reset-gpios = <&pio PE 8 GPIO_ACTIVE_LOW>;
        /* For RTL8211F: PHY datasheet rst time */
        reset-assert-us = <10000>;
        reset-deassert-us = <150000>;
};

&gmac1 {
        phy-mode = "rgmii";
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&gmac1_pins_default>;
        pinctrl-1 = <&gmac1_pins_sleep>;
        phy-handle = <&gmac1_phy0>;
        aw,soc-phy-clk-en;
        sunxi,phy-clk-type = <0>;
        /delete-property/ aw,rgmii-clk-ext;
        tx-delay = <4>;
        rx-delay = <13>;
        status = "okay";
};

&gmac2_phy0 {
        compatible = "ethernet-phy-id0000.0128";
        reset-gpios = <&pio PA 7 GPIO_ACTIVE_LOW>;
        /* For RTL8211F: PHY datasheet rst time */
        reset-assert-us = <10000>;
        reset-deassert-us = <30000>;
        reg = <0x1>;
        max-speed = <100>;
};

&gmac2 {
        phy-mode = "rmii";
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&gmac2_pins_default>;
        pinctrl-1 = <&gmac2_pins_sleep>;
        phy-handle = <&gmac2_phy0>;
        status = "okay";
};

&pio {
        gmac0_pins_default: gmac0@0 {
                pins = "PJ0", "PJ1", "PJ2", "PJ3", "PJ4", "PJ5", "PJ6", "PJ7",
                        "PJ8", "PJ9";
                function = "rgmii0";
                drive-strength = <40>;
                bias-pull-up;
        };

        gmac0_pins_sleep: gmac0@1 {
                pins = "PJ0", "PJ1", "PJ2", "PJ3", "PJ4", "PJ5", "PJ6", "PJ7",
                        "PJ8", "PJ9";
                function = "io_disabled";
                bias-disable;
        };

        gmac1_pins_default: gmac1@0 {
                pins = "PG0", "PG1", "PG2", "PG3", "PG4", "PG6", "PG7",
                        "PG8", "PG9", "PG10", "PG11", "PG12", "PG13", "PG14", "PG15";
                function = "rgmii1";
                drive-strength = <40>;
                bias-pull-up;
        };

        gmac1_pins_sleep: gmac1@1 {
                pins = "PG0", "PG1", "PG2", "PG3", "PG4", "PG6", "PG7",
                        "PG8", "PG9", "PG10", "PG11", "PG12", "PG13", "PG14", "PG15";
                function = "io_disabled";
```

```
                        bias-disable;
                };

                gmac2_pins_default: gmac2@0 {
                        pins = "PD0", "PD1", "PD2", "PD3", "PD4", "PD5", "PD6",
                                "PD7", "PD8", "PD9";
                        function = "rgmii2";
                        drive-strength = <40>;
                        bias-pull-up;
                };

                gmac2_pins_sleep: gmac2@1 {
                        pins = "PD0", "PD1", "PD2", "PD3", "PD4", "PD5", "PD6",
                                "PD7", "PD8", "PD9";
                        function = "io_disabled";
                        bias-disable;
                };

        };
```

注：IDO-EVB5301开发板gmac2与lvds0、mipi硬件复用，如需配置lvds0、mipi，则应先禁用gmac2相关节点。

5.3.LCD

IDO-EVB5301-V1 有一路 MIPI DSI 显示输出接口和两路LVDS输出接口。

dts修改:

mipi-1024x600:

```bash
  backlight0: backlight0 {
    compatible = "pwm-backlight";
    status = "okay";
    brightness-levels = <
      0   1   2   3   4   5   6   7
      8   9   10  11  12  13  14  15
      16  17  18  19  20  21  22  23
      24  25  26  27  28  29  30  31
      32  33  34  35  36  37  38  39
      40  41  42  43  44  45  46  47
      48  49  50  51  52  53  54  55
      56  57  58  59  60  61  62  63
      64  65  66  67  68  69  70  71
      72  73  74  75  76  77  78  79
      80  81  82  83  84  85  86  87
      88  89  90  91  92  93  94  95
      96  97  98  99  100 101 102 103
      104 105 106 107 108 109 110 111
      112 113 114 115 116 117 118 119
      120 121 122 123 124 125 126 127
      128 129 130 131 132 133 134 135
      136 137 138 139 140 141 142 143
      144 145 146 147 148 149 150 151
      152 153 154 155 156 157 158 159
      160 161 162 163 164 165 166 167
      168 169 170 171 172 173 174 175
      176 177 178 179 180 181 182 183
      184 185 186 187 188 189 190 191
      192 193 194 195 196 197 198 199
      200 201 202 203 204 205 206 207
      208 209 210 211 212 213 214 215
      216 217 218 219 220 221 222 223
      224 225 226 227 228 229 230 231
      232 233 234 235 236 237 238 239
      240 241 242 243 244 245 246 247
      248 249 250 251 252 253 254 255>;
    default-brightness-level = <150>;
//    enable-gpios = <&pio PA 9 GPIO_ACTIVE_HIGH>;
    pwms = <&pwmcs0 0 25000 0>;
  };

  panel_0: panel_0@0 {
    compatible = "allwinner,panel-dsi";
    status = "okay";
    enable0-gpios = <&pio PB 7 GPIO_ACTIVE_HIGH>;
    enable-delay-ms = <10>;
    reset-gpios = <&pio PA 6 GPIO_ACTIVE_HIGH>;
    reset-on-sequence = <1 10>, <0 20>, <1 100>;
    reset-off-sequence = <0 100>;
    backlight = <&backlight0>;
    dsi,flags = <(MIPI_DSI_MODE_VIDEO | MIPI_DSI_ASYNC_INCELL)>;
    dsi,lanes = <4>;
    dsi,format = <0>;
    panel-init-sequence = [
      15 00 02 80 5b
      15 00 02 81 78
      15 00 02 82 84
      15 00 02 83 88
      15 00 02 84 88
      15 00 02 85 E3
      15 00 02 86 88
      05 78 01 11
      05 14 01 29
    ];
    panel-exit-sequence = [
      05 00 01 28
      05 78 01 10
    ];

    display-timings {
      native-mode = <&timing0>;
      timing0: timing0 {
        clock-frequency = <51000000>;
        hactive = <1024>;
        vactive = <600>;
        hfront-porch = <160>;
        hsync-len = <10>;
        hback-porch = <160>;
        vfront-porch = <23>;
        vsync-len = <2>;
        vback-porch = <12>;
        hsync-active = <0>;
        vsync-active = <0>;
        de-active = <0>;
        pixelclk-active = <0>;
/*
        clock-frequency = <156408000>;
        hback-porch = <40>;
        hactive = <1200>;
        hfront-porch = <80>;
        hsync-len = <10>;
        vback-porch = <16>;
        vactive = <1920>;
    vfront-porch = <20>;
    vsync-len = <4>;
*/
   };
  };
  port {
   #address-cells = <1>;
   #size-cells = <0>;
   panel0_in: endpoint@0 {
    reg = <0>;
    remote-endpoint = <&panel_output_0>;
   };
  };
 };

 &pio {
   dsi0_4lane_pins_a: dsi0_4lane@0 {
    pins = "PD0", "PD1", "PD2", "PD3", "PD4", "PD5", "PD6", "PD7", "PD8", "PD9";// "PD17", "PD18", "PD19", "PD21";
    function = "dsi";
    drive-strength = <30>;
    bias-disable;
   };
    dsi0_4lane_pins_b: dsi0_4lane@1 {
    pins = "PD0", "PD1", "PD2", "PD3", "PD4", "PD5", "PD6", "PD7", "PD8", "PD9";// "PD17", "PD18", "PD19", "PD21";
    function = "io_disabled";
    bias-disable;
   };
 };

 &dsi0 {
 status = "okay";
 pinctrl-0 = <&dsi0_4lane_pins_a>;
 pinctrl-1 = <&dsi0_4lane_pins_b>;
 #address-cells = <1>;
//  interrupts = <GIC_SPI 83 IRQ_TYPE_LEVEL_HIGH>;
 #size-cells = <0>;
 ports {
  dsi0_out: port@1{
   dsi_out_panel: endpoint {
    remote-endpoint = <&panel_input>;
   };
  };
 };
 panel: panel@0 {
  compatible = "allwinner,virtual-panel";
  status = "okay";
```

```
        reg = <0>;
        ports {
          #address-cells = <1>;
          #size-cells = <0>;
          panel_in: port@0 {
            #address-cells = <1>;
            #size-cells = <0>;
            reg = <0>;
            panel_input: endpoint@0 {
              reg = <0>;
              remote-endpoint = <&dsi_out_panel>;
            };
            panel_input1: endpoint@1 {
            };
          };
          panel_out: port@1 {
            reg = <1>;
            #address-cells = <1>;
            #size-cells = <0>;
            panel_output_0: endpoint@0 {
              reg = <0>;
              remote-endpoint = <&panel0_in>;
            };
            panel_output_1: endpoint@1 {
            };
          };
        };
      };
    };

    &vo0 {
      status = "okay";
    };

    &dlcd0 {
      status = "okay";
    };

    &de {
      chn_cfg_mode = <0>;
      status = "okay";
    };

    &dsi0combophy {
      status = "okay";
    };

    &sunxi_drm{
```

```
    route{
      route_dsi0{
        status = "okay";
      };
    };
  };
```

lvds–1024x600:

```bash
 backlight0: backlight0 {
   compatible = "pwm-backlight";
   status = "okay";
   brightness-levels = <
     0   1   2   3   4   5   6   7
     8   9   10  11  12  13  14  15
     16  17  18  19  20  21  22  23
     24  25  26  27  28  29  30  31
     32  33  34  35  36  37  38  39
     40  41  42  43  44  45  46  47
     48  49  50  51  52  53  54  55
     56  57  58  59  60  61  62  63
     64  65  66  67  68  69  70  71
     72  73  74  75  76  77  78  79
     80  81  82  83  84  85  86  87
     88  89  90  91  92  93  94  95
     96  97  98  99  100 101 102 103
     104 105 106 107 108 109 110 111
     112 113 114 115 116 117 118 119
     120 121 122 123 124 125 126 127
     128 129 130 131 132 133 134 135
     136 137 138 139 140 141 142 143
     144 145 146 147 148 149 150 151
     152 153 154 155 156 157 158 159
     160 161 162 163 164 165 166 167
     168 169 170 171 172 173 174 175
     176 177 178 179 180 181 182 183
     184 185 186 187 188 189 190 191
     192 193 194 195 196 197 198 199
     200 201 202 203 204 205 206 207
     208 209 210 211 212 213 214 215
     216 217 218 219 220 221 222 223
     224 225 226 227 228 229 230 231
     232 233 234 235 236 237 238 239
     240 241 242 243 244 245 246 247
     248 249 250 251 252 253 254 255>;
   default-brightness-level = <150>;
//    enable-gpios = <&pio PB 14 GPIO_ACTIVE_HIGH>;
   pwms = <&pwmcs0 0 25000 0>;
 };

 lvds_panel: lvds_panel@0 {
   compatible = "sunxi-lvds";
   status = "okay";
   backlight = <&backlight0>;
```

```
    bus-format = <MEDIA_BUS_FMT_RGB888_1X7X4_SPWG>;
    // bus-format = <MEDIA_BUS_FMT_RGB888_1X7X4_JEIDA>;
    // bus-format = <MEDIA_BUS_FMT_RGB666_1X7X3_SPWG>;
    reset_gpios = <&pio PA 4 GPIO_ACTIVE_HIGH>;  //复位脚
    display-timings {
      native-mode = <&lvds0_timing0>;
      lvds0_timing0: timing0 {
        clock-frequency = <51287040>;
        hactive = <1024>;
        vactive = <600>;
        hback-porch = <160>;
        hfront-porch = <136>;
        hsync-len = <24>;
        vback-porch = <23>;
        vfront-porch = <12>;
        vsync-len = <1>;

      };
    };
    port {
      lvds_panel_in: endpoint {
        remote-endpoint = <&lvds_panel_out>;
      };
    };
  };
          leds {
                compatible = "gpio-leds";
                pinctrl-names = "default", "sleep";
                status = "okay";

                ……

                lvds_ctl_en_PA9 {
                        gpios = <&pio PA 9 GPIO_ACTIVE_HIGH>;
                        default-state = "on";
                };

                lvds_ctl_pwr {
                        gpios = <&pio PD 20 GPIO_ACTIVE_HIGH>;
                        default-state = "on";
                };

                ……
  };

&pio {
  lvds1_pins_a: lvds1@0 {
    pins = "PD10", "PD11", "PD12", "PD13", "PD14", "PD15", "PD16", "PD17"
, "PD18", "PD19";
    function = "lvds1";
    drive-strength = <30>;
  };

  lvds1_pins_b: lvds1@1 {
    pins = "PD10", "PD11", "PD12", "PD13", "PD14", "PD15", "PD16", "PD17"
, "PD18", "PD19";
    function = "gpio_in";
  };

  leds_gpio1_pins_default: leds@0 {
    pins = ……\
     "PD20", "PA9";
    function = "gpio_out";
  };

  leds_gpio1_pins_sleep: leds@1 {
     pins = ……\
     "PD20", "PA9";
     function = "gpio_in";
  };
};

&lvds0 {
  status = "okay";
  dual-channel = <2>;
  pinctrl-0 = <&lvds1_pins_a>;
  pinctrl-1 = <&lvds1_pins_b>;
  pinctrl-names = "active","sleep";
  ports {
    #address-cells = <1>;
    #size-cells = <0>;
    port@1 {
      #address-cells = <1>;
      #size-cells = <0>;
      reg = <1>;
      lvds_panel_out: endpoint@0 {
        #address-cells = <1>;
        #size-cells = <0>;
        reg = <0>;
        remote-endpoint = <&lvds_panel_in>;
      };
    };
  };
};
```

```
&vo0 {
  status = "okay";
};

&dlcd0 {
  status = "okay";
};

&de {
  chn_cfg_mode = <0>;
  status = "okay";
};

&dsi0combophy {
  status = "okay";
};

&sunxi_drm{
  route{
    route_lvds0{
      status = "okay";
    };
  };
```

5.4.RTC

IDO-EVB5301-V1　采用 HYM8563 作为RTC(*Real Time Clock*)，需要接入 RTC 电池给 RTC 芯片供电才可以保证在短时间系统断电后 RTC 能正常运行。

DTS配置参考: `device/config/chips/t153/configs/bga_demo/linux-5.10-origin/board.dts`

```bash

&twi2 {
        clock-frequency = <400000>;
        pinctrl-0 = <&twi2_pins_default>;
        pinctrl-1 = <&twi2_pins_sleep>;
        pinctrl-names = "default", "sleep";
        /* For stability and backwards compatibility, we recommend setting 'twi_drv_used' to 1 */
        twi_drv_used = <1>;
        /* twi-supply = <&reg_cldo3>; */
        status = "okay";

        pcf8563: rtc@51 {
                compatible = "nxp,pcf8563";
                reg = <0x51>;
        };
};

&pio {
        twi2_pins_default: twi2@0 {
                pins = "PK7", "PK8";
                function = "twi2";
                drive-strength = <10>;
                bias-pull-up;
        };

        twi2_pins_sleep: twi2@1 {
                pins = "PK7", "PK8";
                function = "gpio_in";
        };
};
```

驱动参考：`kernel/linux-5.10-origin/drivers/rtc/rtc-pcf8563.c`

Linux下的rtc使用方法为：

```bash
#同步网络时间
$ ntpdate cn.pool.ntp.org

#查看当前 RTC 的日期和时间:
$ cat /sys/class/rtc/rtc0/date
2021-01-01

$ cat /sys/class/rtc/rtc0/time
17:18:14

$ 查看系统时间
$ date
Wed Mar 12 18:46:59 CST 2025

#设置系统时间
$ date -s "2024-10-09 14:02:30"

#将rtc时间调整为与目前的系统时间一致
$ hwclock -w

#获取硬件rtc当前时间（断电重启读取时间没有太大偏差）
$ hwclock -r
2024-10-09 14:02:35.945604+00:00

#将rtc时间设置为系统时间
hwclock --systohc
```

5.5.UART

DTS配置

RS232 Bash

```
&uart3 {
        pinctrl-names = "default", "sleep";

        pinctrl-0 = <&uart3_pins_active>;

        pinctrl-1 = <&uart3_pins_sleep>;
        status = "okay";
};

&pio {
        uart3_pins_active: uart3_pins@0 {
                pins = "PK9", "PK10";
                function = "uart3";
        };

        uart3_pins_sleep: uart3_pins@1 {
                pins = "PK9", "PK10";
                function = "io_disabled";
        };
};
```

RS485 Bash

```
&uart2 {
        pinctrl-names = "default", "sleep";
        pinctrl-0 = <&uart2_pins_active>;
        pinctrl-1 = <&uart2_pins_sleep>;
        sunxi,uart-485pin_auto =<2>;
        status = "okay";
};

&pio {
        uart2_pins_active: uart2_pins@0 {
                pins = "PB0", "PB1", "PB2";
                function = "uart2";
        };

        uart2_pins_sleep: uart2_pins@1 {
                pins = "PB0", "PB1", "PB2";
                function = "io_disabled";
        };
};
```

配置好串口后，硬件接口对应软件上的节点为：

```bash
UART2:    /dev/ttyAS2
```

收发测试

RS232

```bash
# cat /dev/ttyAS5 &
[1] 453
# echo 123123 > /dev/ttyAS7
123123

# echo 123123 > /dev/ttyAS7
123123

# kill 453
# cat /dev/ttyAS7 &
[2] 454
[1]   Terminated              cat /dev/ttyAS5
# echo 123123 > /dev/ttyAS5
123123

# echo 123123 > /dev/ttyAS5
123123

# echo 123123 > /dev/ttyAS5
123123

#
```

测试RS485收发可以通过以下测试demo进行测试验证

RS485_demo

```c
#include<stdio.h>
#include<stdlib.h>
#include<unistd.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<fcntl.h>
#include<termios.h>
#include<errno.h>
#include<string.h>
#include<pthread.h>
#include<time.h>
#include <linux/serial.h>
#include <termios.h>
#include <sys/ioctl.h>

#define FALSE  -1
#define TRUE   0

volatile int flag = 0;

int uart_open(char* port)
{
    int fd;
    fd = open( port, O_RDWR|O_NOCTTY|O_NDELAY);
    if (fd < 0)
    {
        printf("can't open serial port:%s\n",port);
        return(FALSE);
    }

    if(fcntl(fd, F_SETFL, 0) < 0)
    {
        printf("fcntl failed!\n");
        return(FALSE);
    }

    if(0 == isatty(STDIN_FILENO))
    {
        printf("standard input is not a terminal device\n");
        //return(FALSE);
    }

    return fd;
}

```

```c
void uart_close(int fd)
{
    close(fd);
}

int uart_set(int fd,int speed,int flow_ctrl,int databits,int stopbits,int parity)
{
    int i;
    int status;
    struct serial_rs485 rs485conf;
    int speed_arr[] = {B4000000, B1500000, B115200, B57600, B38400, B19200, B9600, B4800,
                       B2400, B1800, B1200, B600, B300, B200, B150,
                       B134, B110, B75, B50, B0};
    int name_arr[] = {4000000, 1500000, 115200, 57600, 38400, 19200, 9600, 4800, 2400, 1800,
                      1200, 600, 300, 200, 150, 134, 110, 75, 50, 0};

    struct termios options;

    if  ( tcgetattr( fd,&options)  !=  0)
    {
        perror("setupSerial 1");
        return FALSE;
    }

    for ( i= 0;  i < sizeof(speed_arr) / sizeof(int);  i++)
    {
        if  (speed == name_arr[i])
        {
            cfsetispeed(&options, speed_arr[i]);
            cfsetospeed(&options, speed_arr[i]);
        }
    }

    // RS485
    if (ioctl(fd, TIOCGRS485, &rs485conf) < 0) {
        return FALSE;
    }
    rs485conf.flags |= SER_RS485_ENABLED;
    if (ioctl(fd, TIOCSRS485, &rs485conf) < 0) {
        return FALSE;
    }

    options.c_cflag |= CLOCAL;  //修改控制模式，保证程序不会占用串口
    options.c_cflag |= CREAD;   //修改控制模式，使得能够从串口中读取输入数据
    switch(flow_ctrl)   //设置数据流控制
    {

        case 0 :
            options.c_cflag &= ~CRTSCTS;   //不使用流控制
            //options.c_iflag &= ~(IXON | IXOFF | IXANY);
            break;

        case 1 :
            options.c_cflag |= CRTSCTS; //使用硬件流控制
            break;
        case 2 :
            options.c_cflag |= IXON | IXOFF | IXANY;//使用软件流控制
            break;
    }


    options.c_cflag &= ~CSIZE;//设置数据位   //屏蔽其他标志位

    switch (databits)
    {
        case 5:
            options.c_cflag |= CS5;
            break;
        case 6:
            options.c_cflag |= CS6;
            break;
        case 7:
            options.c_cflag |= CS7;
            break;
        case 8:
            options.c_cflag |= CS8;
            break;
        default:
            fprintf(stderr,"Unsupported data size\n");
            return FALSE;
    }

    switch (parity)  //设置校验位
    {
        case 'n':
        case 'N': //无奇偶校验位。
            options.c_cflag &= ~PARENB;
            options.c_iflag &= ~INPCK;
            break;
        case 'o':
        case 'O'://设置为奇校验
            options.c_cflag |= (PARODD | PARENB);
            options.c_iflag |= INPCK;
            break;
        case 'e':
        case 'E'://设置为偶校验
            options.c_cflag |= PARENB;
            options.c_cflag &= ~PARODD;
            options.c_iflag |= INPCK;
            break;
        case 's':
        case 'S': //设置为空格
            options.c_cflag &= ~PARENB;
            options.c_cflag &= ~CSTOPB;
            break;
        default:
            fprintf(stderr,"Unsupported parity\n");
            return FALSE;
    }

    switch (stopbits)  // 设置停止位
    {
        case 1:
            options.c_cflag &= ~CSTOPB; break;
        case 2:
            options.c_cflag |= CSTOPB; break;
        default:
            fprintf(stderr,"Unsupported stop bits\n");
            return FALSE;
    }

    options.c_iflag &= ~ (IXON | IXOFF | IXANY);
    options.c_iflag &= ~ (INLCR | ICRNL | IGNCR);
    options.c_oflag &= ~(ONLCR | OCRNL);

    options.c_lflag  &= ~(ICANON | ECHO | ECHOE | ISIG);
    options.c_oflag  &= ~OPOST;

    //设置等待时间和最小接收字符
    options.c_cc[VTIME] = 1; /* 读取一个字符等待1*(1/10)s */
    options.c_cc[VMIN] = 0; /* 读取字符的最少个数为1 */

    //如果发生数据溢出，接收数据，但是不再读取 刷新收到的数据但是不读
    // tcflush(fd,TCIFLUSH);

    //激活配置 （将修改后的termios数据设置到串口中）
    if (tcsetattr(fd,TCSANOW,&options) != 0)
    {
        perror("com set error!\n");
        return FALSE;
```

```c
    }
    return TRUE;
}
int uart_init(int fd, int speed,int flow_ctrl,int databits,int stopbits,int parity)
{
    int err;

    if (uart_set(fd,speed,flow_ctrl,databits,stopbits,parity) == FALSE)  //设置串口数据帧格式
    {
        return FALSE;
    }  else  {
        return  TRUE;
    }
}


int uart_recv(int fd, char *rcv_buf,int data_len)
{
    int len,fs_sel;
    fd_set fs_read;

    struct timeval time;

    FD_ZERO(&fs_read);
    FD_SET(fd,&fs_read);

    time.tv_sec = 1;
    time.tv_usec = 0;

    fs_sel = select(fd+1,&fs_read,NULL,NULL,&time);
    if(fs_sel > 0)
    {
        len = read(fd,rcv_buf,data_len);
        return len;
    }else if (fs_sel == 0){
        return 0;
    }
}

int uart_send(int fd, char *send_buf,int data_len)
{
    int len = 0;

    len = write(fd,send_buf,data_len);
    if (len == data_len )
    {
        return len;
    }  else {
        tcflush(fd,TCOFLUSH);
        return FALSE;
    }
}
static void * uart_tx_pthread(void *arg)
{
    char test_cmd[] = {0x31, 0x32, 0x33};

    uart_send(*(int*)arg, test_cmd, sizeof(test_cmd));
}

static void * uart_rx_pthread(void *arg)
{
    char recv_buf[128];
    int len = 0;
    int i;
    int ret;

    memset(recv_buf, 0, sizeof(recv_buf));
    while (1) {
        ret = uart_recv(*(int*)arg, recv_buf+len, sizeof(recv_buf));
        if (ret > 0){
            len += ret;
            printf(" len=%d\n", len);
        }else if (ret == 0){
            printf("timeout \n");
            printf("recv:");
            for (i=0; i<len; i++)
            {
                printf("%02x \n",recv_buf[i]);
            }
            printf("\n");
            break;
        }
        usleep(100000);
    }

    if ( (recv_buf[0] == 0x31) && (recv_buf[1] == 0x32) && (recv_buf[2] == 0x33) )
    {
        printf("check okay\n");
        flag = 1;
    }else{
        printf("check fail\n");
        flag = -0;
    }
```

```c
}

int main(int argc, char **argv)
{
    int fd_r;
    int fd_w;
    int err;
    int baud;
    int databit;
    int stopbit;
    char parity;

    if (argc != 3){
        printf("usage: ./rs485_test /dev/ttyS2 /dev/ttyAS4\n");
        return -1;
    }

    fd_r = uart_open(argv[1]);
    if (fd_r < 0) {
        printf("open %s fail!", argv[1]);
        return fd_r;
    }

    fd_w = uart_open(argv[2]);
    if (fd_w < 0) {
        printf("open %s fail!", argv[2]);
        close(fd_r);
        return fd_w;
    }
    baud = 9600;
    databit = 8;
    stopbit = 1;
    parity = 'N';

    err = uart_init(fd_r,baud,0,databit, stopbit, parity);
    if (err == FALSE){
        printf("uart init err\n");
        return -1;
    }

    err = uart_init(fd_w,baud,0,databit, stopbit, parity);
    if (err == FALSE){
        printf("uart init err\n");
        return -1;
    }

    pthread_t pt1, pt2;
    if (pthread_create(&pt1, NULL, uart_rx_pthread, (void *)&fd_r) == -1) {
        printf("pthread_create failed\n");
        return -1;
    }
    usleep(10000);

    if (pthread_create(&pt2, NULL, uart_tx_pthread, (void *)&fd_w) == -1) {
        printf("pthread_create failed\n");
        return -1;
    }

    if (pthread_join(pt2, NULL))
    {
        printf("thread is not exit...\n");
        return -2;
    }

    if (pthread_join(pt1, NULL))
    {
        printf("thread is not exit...\n");
        return -2;
    }

    close(fd_r);
    close(fd_w);

    if (flag == 1){
        return 0;
    }else{
        return -1;
    }
}
```

6.开发案例

6.1.Qt5使用

详情参考

百度网盘：EVB5301-Linux/3.开发手册/《IDO-EVB5301-V1 QT应用开发手册》.pdf

6.2.MQTT使用

详情参考

百度网盘：EVB5301-Linux/3.开发手册/《IDO-EVB5301-V1 MQTT应用案例》.pdf

6.3.GDB使用

详情参考

百度网盘：EVB5301-Linux/3.开发手册/《IDO-EVB5301-V1 GDB开发手册》.pdf

6.4. LVGL使用

详情参考

百度网盘：EVB5301-Linux/3.开发手册/《IDO-EVB5301-V1 LVGL应用开发手册》.pdf