IDO-IPC8815-V1 Linux开发手册

Industio 触觉智能

IDO-IPC8815-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

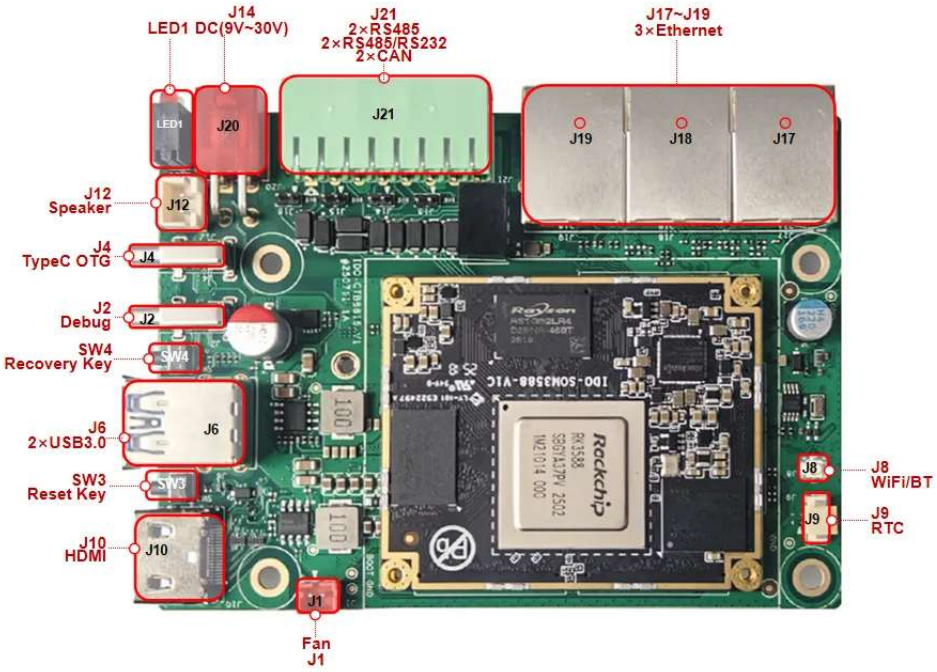
版本	PCBA版本	修订内容	修订	审核	日期
V1.0	V1D	创建文档	MHK	IDO	2026/01/16

1. 上手指南

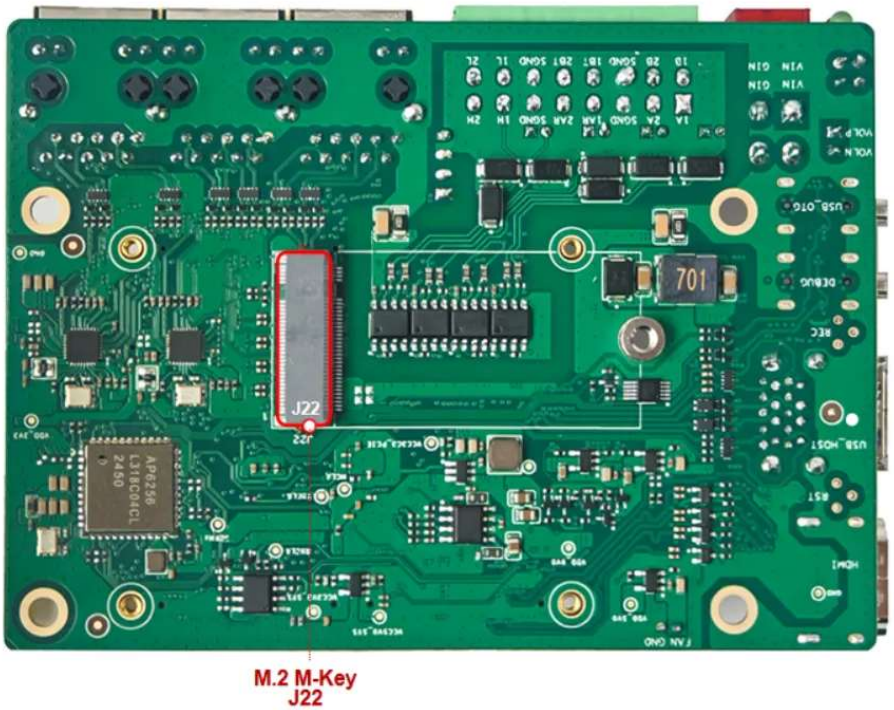
1.1. 介绍

IDO-IPC8815 配置 Rockchip 八核 64 位 AIOT 处理器，采用了 8nm LP 制程；搭载八核（Cortex-A76 x 4 + Cortex-A55 x 4）64位 CPU，主频高达2.4 GHz。

集成 ARM Mali-G610 MP4 四核 GPU，内置 AI 加速器 NPU，可提供6 Tops 算力，支持主流的深度学习框架；最大支持32 GB 大内存；支持 8K 视频编解码和多种格式的视频输入输出；支持多 种操作系统；可适用于 ARM PC、边缘计算、云服务器、智能NVR 等领域。



IDO-IPC8815-V1D 正面



IDO-IPC8815-V1D 背面

2. Linux开发

2.1. 获取 SDK

下载路径：百度网盘/Linux/软件资料/SDK/

2.2. 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~/IPC8815为例

```Bash
$ mkdir ~/IPC8815
$ cd ~/IPC8815
$ tar -xvf rk3588_linux6.1_rkr6_v1.tar.gz -C ./

$ cd rk3588_linux6.1_rkr6_v1

#查看分支(默认分支为ido-ctb8815-ubuntu)
$ git branch
  ido-ctb8815-debian
  ido-ctb8815-debian-preempt-rt
* ido-ctb8815-ubuntu
  ido-ctb8815-ubuntu-preempt-rt

#获取代码
$ git reset --hard

```

2.3. SDK目录介绍

```bash
$ tree -L 1
.
├── app
├── buildroot
├── build.sh -> device/rockchip/common/scripts/build.sh
├── common -> device/rockchip/common
├── Copyright_Statement.md -> docs/licenses/LICENSE
├── debian
├── debian_rootfs
├── device
├── docs
├── external
├── kernel -> kernel-6.1
├── kernel-6.1
├── Makefile -> device/rockchip/common/Makefile
├── output
├── prebuilts
├── README.md -> device/rockchip/common/README.md
├── rkbin
├── rkflash.sh -> device/rockchip/common/scripts/rkflash.sh
├── rockdev -> output/firmware
├── tools
├── u-boot
└── yocto
```

```bash
app：存放上层应用 APP，主要是一些应用Demo。
buildroot：基于 Buildroot（2024）开发的根文件系统。
device/rockchip：存放芯片板级配置以及SDK编译和打包固件的脚本和文件等。
docs：存放通用开发指导文档、芯片平台相关文档、Linux 系统开发相关文档、其他参考文档等。
external：存放第三方相关仓库，包括显示、音视频、摄像头、网络、安全等。
kernel：存放 Kernel 开发的代码。
output：存放每次生成的固件信息、编译信息、XML、主机环境等。
prebuilts：存放交叉编译工具链。
rkbin：存放 Rockchip 相关二进制和工具。
rockdev：存放编译输出固件,实际软链接到 output/firmware 。
tools：存放 Linux 和 Window 操作系统下常用工具。
u-boot：存放基于 v2017.09 版本进行开发的 U-Boot 代码。
yocto: 存放yocto相关编译脚本和代码。
```

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）

2.4. 配置文件介绍

在 `device/rockchip/rk3588` 目录下，有不同板型的配置文件(xxxx_defconfig)，用于管理 SDK 每个环节的编译配置。rockchip_rk3588_ido_IPC8815_v1b*_defconfig 作为基础的配置。其他版型的配置文件会引用 rockchip_rk3588_ido_IPC8815_v1b*_defconfig 并且覆盖其中的一些变量配置。 以ido-evb3562-v2a-mipi-800x1280-buildroot_defconfig 做介绍：

```Bash
RK_BUILDROOT_BASE_CFG="rk3588"
RK_KERNEL_DTS_NAME="ido-ctb8815-v1a"                    # 板型 dts 名称
RK_KERNEL_CFG="rockchip_linux_defconfig"                # kernel 编译配置
RK_PARAMETER="parameter.txt"
RK_UBOOT_SPL=y
RK_USE_FIT_IMG=y
RK_ROOTFS_SYSTEM_UBUNTU22=y
```

2.5. 分区说明

parameter.txt 文件中包含了固件的分区信息，如下所示：

```Bash
FIRMWARE_VER: 1.0
MACHINE_MODEL: RK3588
MACHINE_ID: 007
MANUFACTURER: RK3588
MAGIC: 0x5041524B
ATAG: 0x00200800
MACHINE: 0xffffffff
CHECK_MASK: 0x80
PWR_HLD: 0,0,A,0,1
TYPE: GPT
GROW_ALIGN: 0
CMDLINE: mtdparts=:0x00002000@0x00004000(uboot),0x00002000@0x00006000(misc),0x00020000@0x00008000(boot),0x00040000@0x00028000(recovery),0x00010000@0x00068000(backup),0x00400000@0x00078000(userdata),0x00040000@0x00478000(oem),-@0x004B8000(rootfs:grow)
uuid:rootfs=614e0000-0000-4b53-8000-1d28000054a9
uuid:boot=7A3F0000-0000-446A-8000-702F00006273
```

CMDLINE 属性是我们关注的地方，以 uboot 为例， 0x00002000@0x00004000(uboot) 中 0x00002000 为uboot 分区的起始位置，0x00004000 为分区的大小，以此类推。

3. 编译

3.1. 选择项目

```Bash
mhkwork@dell-PowerEdge-R430:~/work/rk/rk3588/linux/rk3588_linux6.1/rk3588_linux6.1_rkr6_250715$ ./build.sh lunch
Log colors: message notice warning error fatal

Log saved at /mnt/5c953a98-9ee6-45b9-8cea-a2898eb313d7/mhk/rk/rk3588/linux/rk3588_linux6.1/rk3588_linux6.1_rkr6_250715/output/sessions/2026-01-16_18-38-07
Pick a defconfig:

1. rockchip_ido_ctb8815_v1_hdmi_ubuntu20_defconfig
2. rockchip_ido_ctb8815_v1_hdmi_ubuntu22_defconfig
```

选择对应的数字即可。

3.2. 自动编译

```Bash
$ ./build.sh
```

固件编译后生成路径：rockdev/update.img

3.3. 分步编译

3.3.1. 编译uboot

```Bash
./build.sh uboot
```

固件编译后生成路径：rockdev/uboot.img

3.3.2. 编译kernel

```Bash
./build.sh kernel
```

固件编译后生成路径：rockdev/boot.img

3.3.3. 编译ubuntu20镜像

```Bash
./build.sh ubuntu20
```

3.3.4. 编译ubuntu22镜像

```Bash
./build.sh ubuntu22
```

3.3.5. 编译debian镜像

```Bash
./build.sh debian
```

3.3.6. 编译recovery

```Bash
./build.sh recovery
```

固件编译后生成路径：rockdev/recovery.img

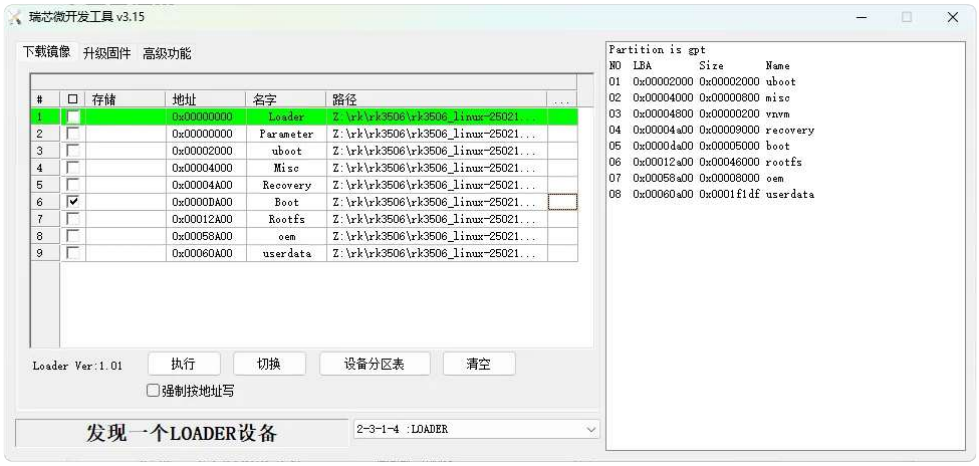
3.3.7. 固件打包

```Bash
./build.sh updateimg
```

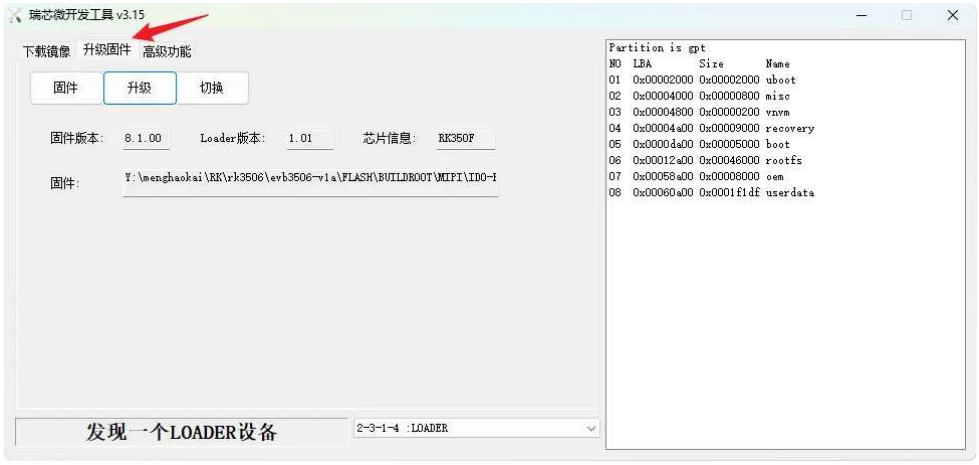
4. 烧录说明

4.1. 整包烧录

4.1.1. 运行RK开发工具



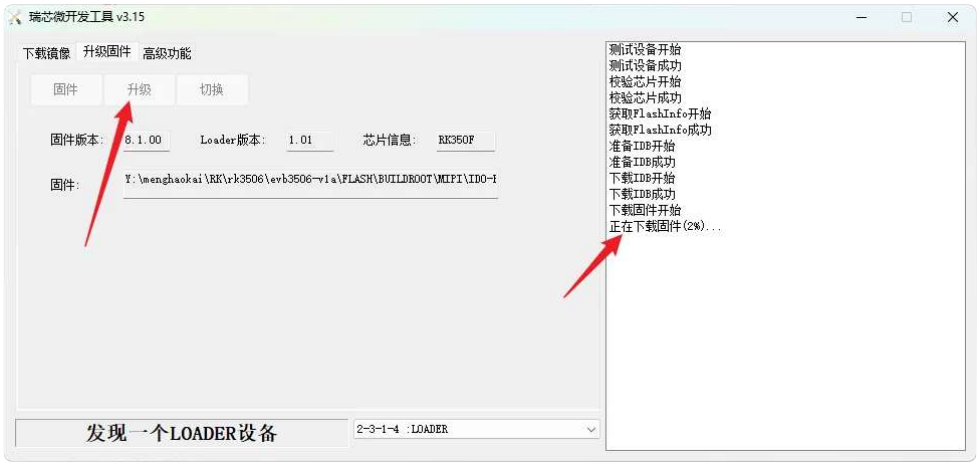
4.1.2. 点击升级固件



4.1.3. 点击固件

选择：rockdev/update.img

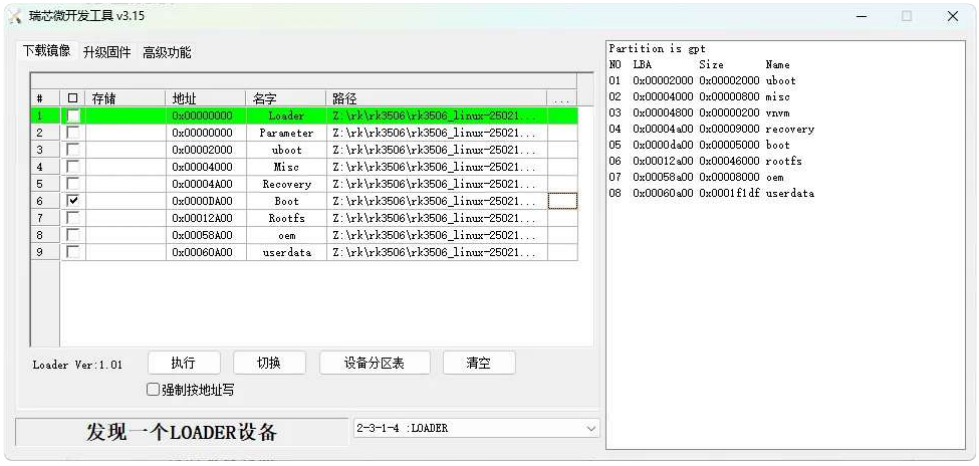
4.1.4. 点击升级

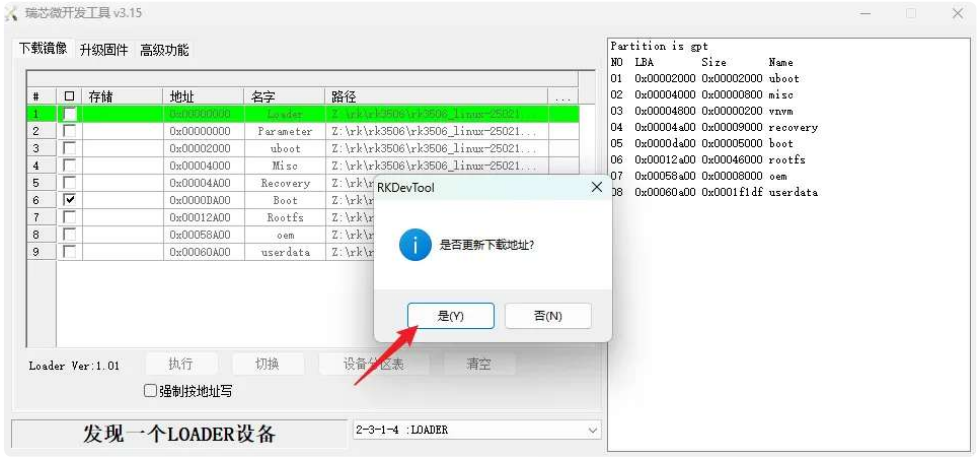


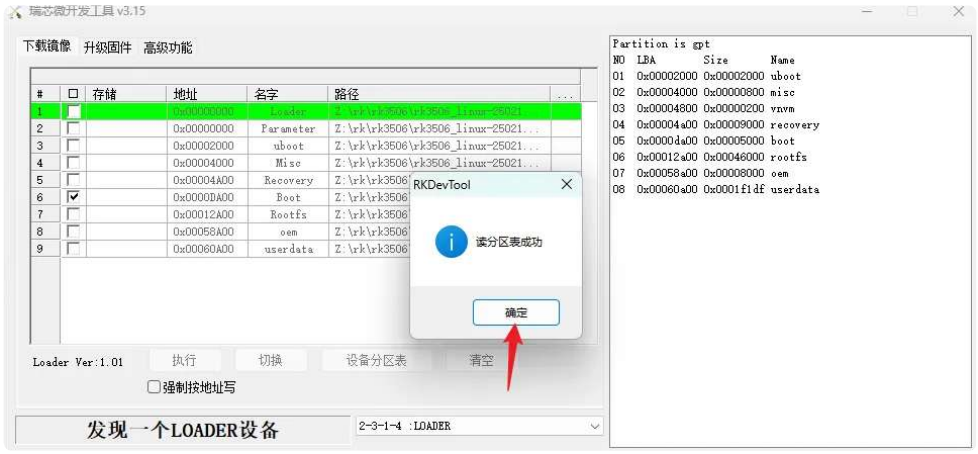
点击升级后，右侧会开始下载固件；下载完成后，板子会自动启动。

4.2. 分包烧录

4.2.1. 更新下载地址

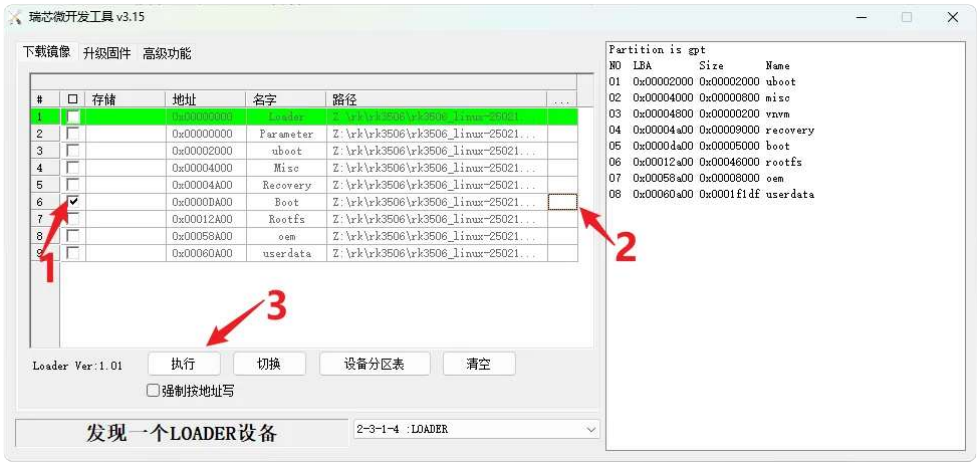






4.2.2. 烧录

1. 在对应的分区前选择对号
2. 然后点击右侧去选择对应的镜像
3. 点击执行即可烧录



5. 驱动开发

5.1. Ethernet

5.1.1.1. 公共配置

kernel/arch/arm64/boot/dts/rk3588s.dtsi

```
gmac1: ethernet@fe1c0000 {
  compatible = "rockchip,rk3588-gmac", "snps,dwmac-4.20a";
  reg = <0x0 0xfe1c0000 0x0 0x10000>;
  interrupts = <GIC_SPI 234 IRQ_TYPE_LEVEL_HIGH>,
         <GIC_SPI 233 IRQ_TYPE_LEVEL_HIGH>;
  interrupt-names = "macirq", "eth_wake_irq";
  rockchip,grf = <&sys_grf>;
  rockchip,php_grf = <&php_grf>;
  clocks = <&cru CLK_GMAC_125M>, <&cru CLK_GMAC_50M>,
     <&cru PCLK_GMAC1>, <&cru ACLK_GMAC1>,
     <&cru CLK_GMAC1_PTP_REF>;
  clock-names = "stmmaceth", "clk_mac_ref",
          "pclk_mac", "aclk_mac",
          "ptp_ref";
  resets = <&cru SRST_A_GMAC1>;
  reset-names = "stmmaceth";
  power-domains = <&power RK3588_PD_GMAC>;

  snps,mixed-burst;
  snps,tso;

  snps,axi-config = <&gmac1_stmmac_axi_setup>;
  snps,mtl-rx-config = <&gmac1_mtl_rx_setup>;
  snps,mtl-tx-config = <&gmac1_mtl_tx_setup>;
  status = "disabled";

  mdio1: mdio {
    compatible = "snps,dwmac-mdio";
    #address-cells = <0x1>;
    #size-cells = <0x0>;
  };

  gmac1_stmmac_axi_setup: stmmac-axi-config {
    snps,wr_osr_lmt = <4>;
    snps,rd_osr_lmt = <8>;
    snps,blen = <0 0 0 0 16 8 4>;
  };

  gmac1_mtl_rx_setup: rx-queues-config {
    snps,rx-queues-to-use = <1>;
    queue0 {};
  };

  gmac1_mtl_tx_setup: tx-queues-config {
    snps,tx-queues-to-use = <1>;
```

```
        queue0 {};
      };
    };
```

板级的配置

```Bash
&gmac1 {
  /* Use rgmii-rxid mode to disable rx delay inside Soc */
  //phy-mode = "rgmii-rxid";
  phy-mode = "rgmii";
  clock_in_out = "input";

  snps,reset-gpio = <&gpio1 RK_PD7 GPIO_ACTIVE_LOW>;
  snps,reset-active-low;
  /* Reset time is 20ms, 100ms for rtl8211f */
  snps,reset-delays-us = <0 20000 100000>;

  pinctrl-names = "default";
  pinctrl-0 = <&gmac1_miim
     &gmac1_tx_bus2
     &gmac1_rx_bus2
     &gmac1_rgmii_clk
     &gmac1_rgmii_bus
     &gmac1_clkinout>;

  tx_delay = <0x31>;
  rx_delay = <0x3a>;

  phy-handle = <&rgmii_phy>;
  status = "okay";
};

```

5.3.显示

5.1.1. HDMI

```Bash
&hdptxphy_hdmi0 {
    status = "okay";
};

&route_hdmi0 {
      status = "okay";
    connect = <&vp0_out_hdmi0>;
};

&hdmi0 {
    enable-gpios = <&gpio4 RK_PA7 GPIO_ACTIVE_HIGH>;
    status = "okay";
};

&hdmi0_in_vp0 {
    status = "okay";
};

&hdmi0_in_vp1 {
    status = "disabled";
};

//&vp0 {
//    assigned-clocks = <&cru DCLK_VOP0>;
//    assigned-clock-parents = <&hdptxphy_hdmi_clk0>;
//};

&hdmi0_sound {
    status = "okay";
};

```

5.4.RTC

IDO-IPC8815-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：`kernel/arch/arm64/boot/dts/ido-ctb8815-v1a.dts`

驱动参考：`kernel/drivers/rtc/rtc-hym8563.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
2025-05-27

$ cat /sys/class/rtc/rtc0/time
10:10:30

$ 查看系统时间
$ date
Tue May 27 18:10:41 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

```bash
&uart6 {
    status = "okay";
    pinctrl-names = "default";
    pinctrl-0 = <&uart6m1_xfer>;
    rs485-txen-gpio = <&gpio1 RK_PA2 GPIO_ACTIVE_HIGH>;
};
```

配置好串口后，硬件接口对应软件上的节点为：

```
UART6:   /dev/ttyS6
```

收发测试

使用microcom可以进行串口收发测试，命令如下：

```
root@rk3588:~# microcom -s 115200 -p /dev/ttyS6
```

注意：测试完成，按Ctrl+x退出。