Rockchip RK3576 USB 开发指南

文件标识：RK-SM-YF-C01

发布版本：V1.1.0

日期：2024-10-09

文件密级：□绝密 □秘密 □内部资料 ■公开

瑞芯微电子股份有限公司

Rockchip Electronics Co., Ltd.

地址： 福建省福州市铜盘路软件园A区18号

网址： www.rock-chips.com

客户服务电话： +86-4007-700-590

客户服务传真： +86-591-83951833

客户服务邮箱： fae@rock-chips.com

前言

概述

本文档提供 RK3576 USB 模块的开发指南，目的是让开发者理解 RK3576 USB 控制器和 PHY 的硬件电路设计和软件 DTS 配置，以便开发者根据产品的 USB 应用需求进行灵活设计和快速开发。

芯片名称	内核版本
RK3576	Linux-6.1

读者对象

本文档（本指南）主要适用于以下工程师：

技术支持工程师

软件开发工程师

硬件开发工程师

修订记录

日期	版本	作者	修改说明
2024-04-24	V1.0.0	吴良峰 王明成	初始版本
2024-05-30	V1.0.1	吴良峰	修正 usbdp_phy 节点名称
2024-10-09	V1.1.0	吴良峰	修正 usb 控制器节点名称 修改 Type-C 控制器芯片支持列表，新增支持 AW35615

目录

1. RK3576 USB 控制器和 PHY 简介

RK3576 支持2 个 独立的 USB 3.1 OTG 控制器、2 个 独立的 USB 2.0 PHY、1 个 USB 3.1/DP Combo PHY 和 1 个 USB 3.1/SATA/PCIe Combo PHY。如果要了解更详细的 USB 控制器特性，请参阅 RK3576 TRM。

RK3576 USB 新增如下功能：

1. OTG0/1 都支持 MMU，允许 USB 控制器硬件访问超过 4G 的内存空间；
2. OTG1 支持 CCI 保证 Cache 一致性，即不需要 CPU 执行刷 cache 的操作（OTG0 不支持）；
3. OTG0 DP/DM 支持与 UART/JTAG 复用，SDK 默认使能该功能，需要结合 RK USB to DEBUG 转接板使用。

Note:

1. USB 3.1 Gen1 物理层传输速率为 5Gbps，USB 2.0 物理层传输速率为 480Mbps;
2. USB 3.1/DP Combo PHY 支持 4 x lanes，可以同时支持 USB 3.1 + DP 2 x lanes；
3. 使用限制：USB 3.1/SATA/PCIe Combo PHY 在同一时刻，只能支持一种工作模式，也即 USB 3.1 与 SATA/PCIe 接口是互斥的。

表 1 RK3576 USB 控制器和 PHY 的连接关系

USB 接口名称(原理图)	**USB 控制器**	**USB PHY**
USB OTG0	OTG0 (DWC3&xHCI)	USB3.1/DP ComboPHY + USB2.0 PHY0
USB OTG1	OTG1 (DWC3&xHCI)	USB3.1/SATA/PCIe ComboPHY1 + USB2.0 PHY1

RK3576 USB 控制器和芯片端 USB 传输数据的 pin 脚的对应关系如下表 2 所示。

表 2 RK3576 USB 控制器和 USB pin 脚的对应关系

USB控制器/Pin脚	**RK3576 USB data pin**
USB 3.1 OTG0	USB2_OTG0_DP/USB2_OTG0_DM, USB3_OTG0_SSRX1P/USB3_OTG0_SSRX1N, USB3_OTG0_SSTX1P/USB3_OTG0_SSTX1N, USB3_OTG0_SSRX2P/USB3_OTG0_SSRX2N, USB3_OTG0_SSTX2P/USB3_OTG0_SSTX2N,
USB 3.1 OTG1	USB2_OTG1_DP/USB2_OTG1_DM, USB3_OTG1_SSTXP/USB3_OTG1_SSTXN, USB3_OTG1_SSRXP/USB3_OTG1_SSRXN,

RK3576 USB 控制器和 PHY 的内部连接关系，以及对应的常见 USB 物理接口如下图 1 所示。

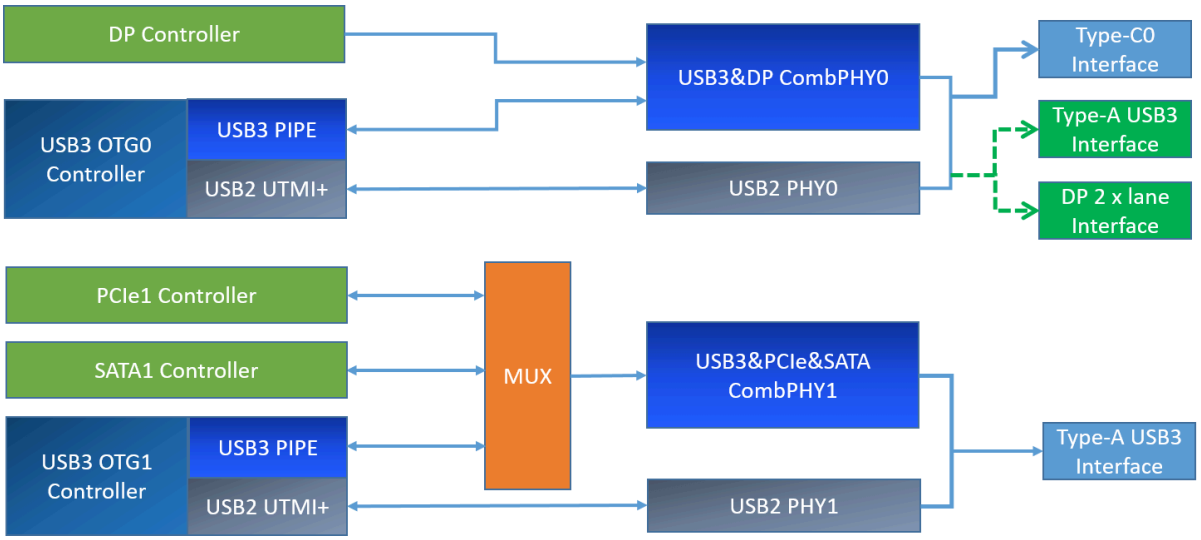

图 1 RK3576 USB 控制器和 PHY 的连接示意图

Note：

1. RK3576 USB 支持的接口类型并不局限于图 1 所描述的 Type-C/A USB 接口类型，还可以支持所有常见的 USB 接口，包括 Type-C USB 2.0/3.1，Type-A USB 2.0/3.1，Micro USB 2.0/3.1 等。为了适配不同的 USB 电路设计和接口类型，Linux-6.1 内核 USB 驱动已经做了软件兼容，开发者只需要根据产品的 USB 硬件电路，对 Linux USB DTS 进行正确配置，即可使能对应的 USB 接口功能。详细的 USB DTS 配置方法，请参考 RK3576 USB DTS 配置。
2. **特殊使用限制：**OTG1/PCIe1/SATA1 控制器访问总线互斥，如果硬件电路设计上已使用了PCIe1或者SATA1，则 OTG1 USB2 和 USB3 功能都无法使用。

2. RK3576 USB Config Map

RK3576 的 2 个独立的 USB 控制器可以支持如下图 2 ~ 4 所列出的配置方式构成不同的产品形态。

USB OTG0 可以支持 5 种硬件电路设计：

Config0: Type-C0 USB3.1 OTG0 with DP function

Config1: USB 2.0 OTG0 + DP 4 x Lane (Swap off)

Config2: USB 2.0 OTG0 + DP 4 x Lane (Swap on)

Config3: USB 3.1 OTG0 + DP 2 x Lane (Swap on)

Config4: USB 3.1 OTG0 + DP 2 x Lane (Swap off)

USB OTG1 支持的 2 种硬件电路设计：

Config0: USB 2.0 only OTG1

Config1: USB 3.1 OTG1

如果要了解更详细的 USB 配置表，请参考 RK3576 SDK EVB 参考原理图章节 USB/DP Configure Map。

Note：

1. USB 2.0/3.1 OTG1 都不能与 PCIe1/SATA1 同时使用
2. USBDP Swap on/off 两种配置的 Lanes 对应关系如下：

Swap off：Lane0/1/2/3 TxData mapping to Lane0/1/2/3_TXDP/N

Swap on：Lane0/1/2/3 TxData mapping to Lane2/3/0/1_TXDP/N

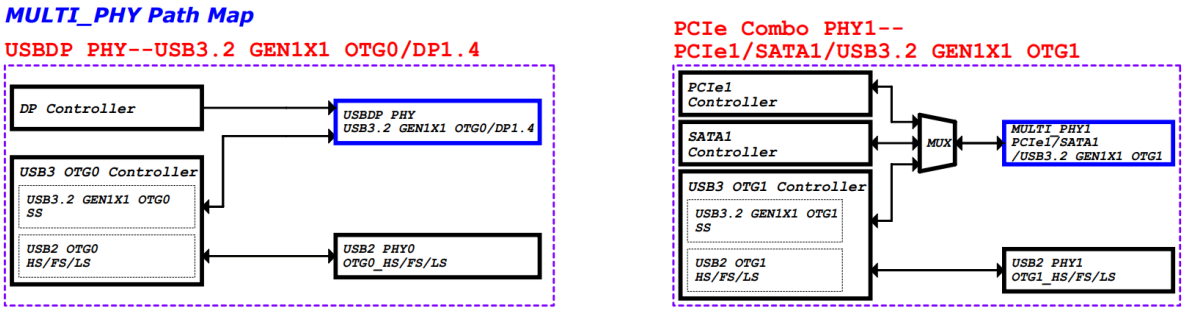

图 2 RK3576 USB Path Map

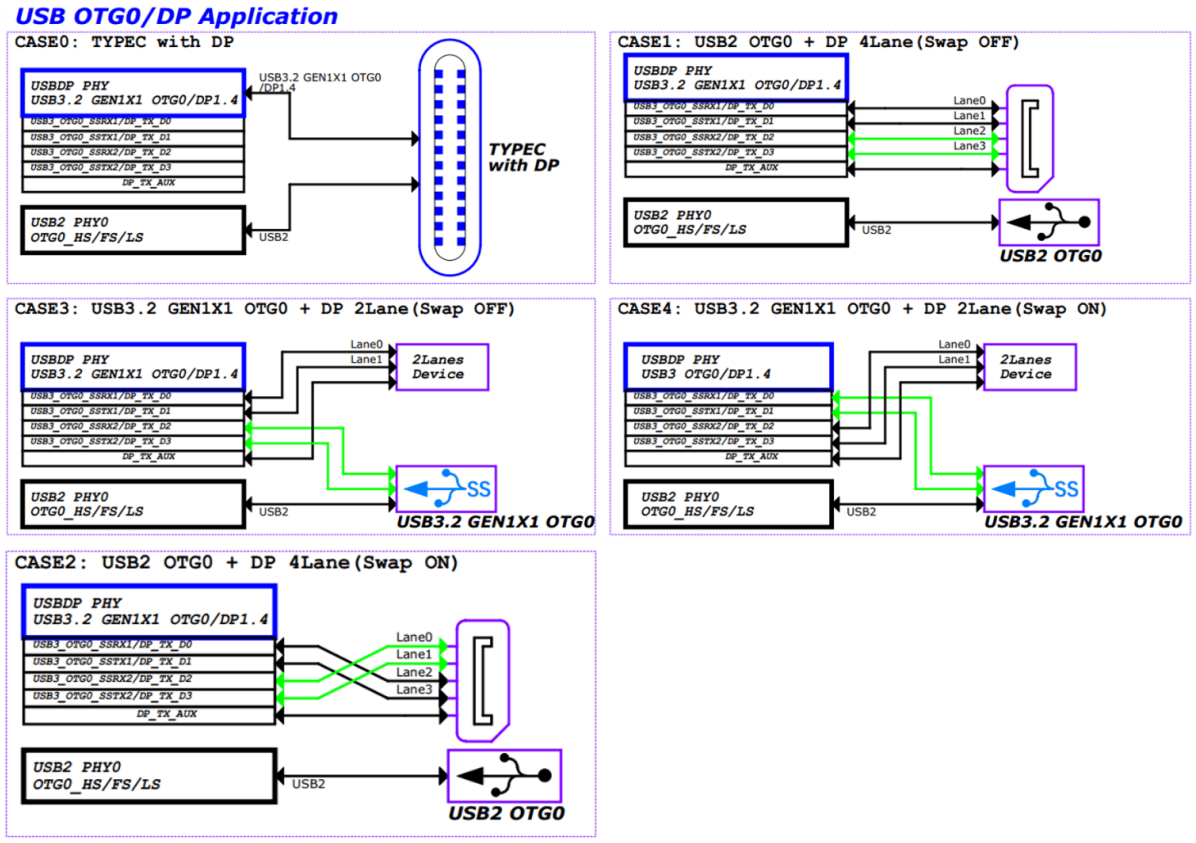

图 3 RK3576 OTG0 USBDP Path Map

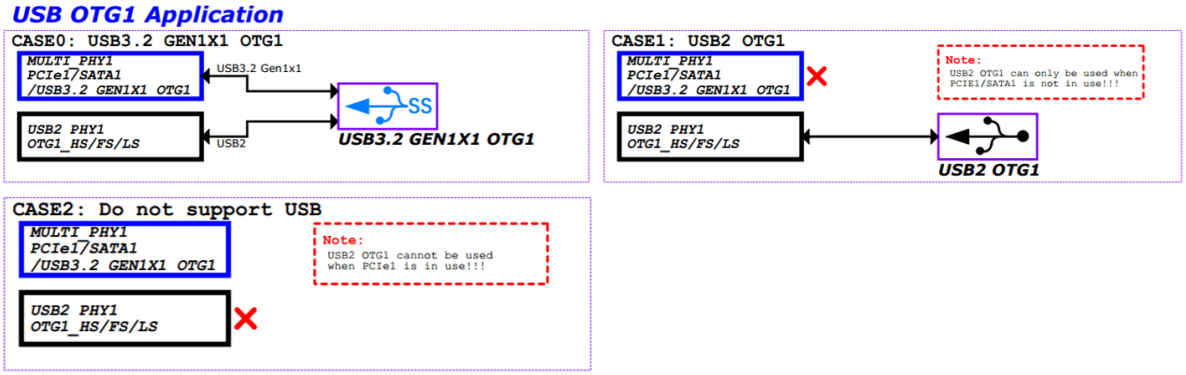

图 4 RK3576 OTG1 MULTI PHY Path Map

3. RK3576 USB 硬件电路设计

本章节主要说明 RK3576 USB 在实际应用中，可支持的各种硬件电路设计方案。RK3576可以支持的接口如下：

- USB30/DP1.4 MULTI0
- USB30/PCIE2.0/SATA30 MULTI1

3.1 USB 控制器供电及功耗管理

RK3576 USB 控制器的供电电源是 VD_LOGIC。同时，芯片内部有设计 USB 控制器专用的 power domain：USB3.1 OTG0 位于 PD_USB；USB3.1 OTG1 位于 PD_PHP。

在实际使用场景中，Linux USB 控制器驱动会根据 USB 接口的工作情况，基于 Linux PM Runtime 机制，动态开关 USB 控制器的 PD，以降低 USB 控制器的功耗。而当系统进入二级待机时，为了达到最优功耗的目的，软件会强制关闭 USB 控制器的所有 PD。因此，在实际产品的应用场景中，如果需要在二级待机时，保持 USB 控制器的寄存器工作状态，则需要在 USB 控制器驱动中调用函数 `device_init_wakeup`，避免二级待机时关闭 USB 控制器的 PD。

USB 控制器的功耗控制策略如下：

1. 对于不使用的 USB 控制器，需要将对应的控制器 DTS 节点配置为 disabled；
2. 对于内核已启用的 USB 控制器，内核 USB 驱动已经支持 USB 控制器 Auto suspend 功能 (当 USB HOST 接口不接任何外设时，控制器自动进入 suspend 低功耗状态)，因此，开发者不需要对 USB 控制器的动态功耗管理进行调试。

3.2 USB PHY 供电及功耗管理

3.2.1 USB 2.0 PHY 供电及功耗管理

RK3576 支持 2 个 独立的 USB 2.0 PHY。在芯片内部，所有 USB 2.0 PHY 都属于 VD_USBPHY，并且，所有 USB 2.0 PHY 共用如下图 5 所示的 3 路外部供电电源。因此，在系统运行时，无法通过硬件断电和关闭 PD 的简单方法，来降低 USB 2.0 PHY 的功耗。

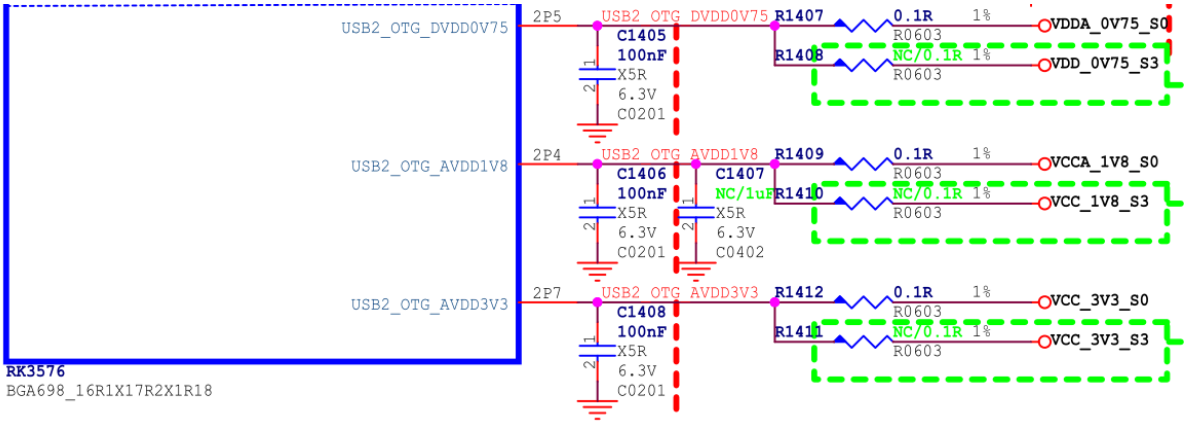

图 5 USB 2.0 PHY 供电电源

需要注意的是，在实际电路中，USB 2.0 PHY 的供电电压值超过规定的最大值或者低于规定的最小值，可能会导致 USB 连接异常。

表 3 USB 2.0 PHY 供电电压要求

供电电源	最小	正常	最大	Unit
USB2_OTG_DVDD_0V75	0.6975	0.75	0.825	V
USB2_OTG_AVDD_1V8	1.674	1.8	1.98	V
USB2_OTG_AVDD_3V3	3.069	3.3	3.63	V

USB 2.0 PHY 的功耗控制策略如下：

1. 为了支持 Maskrom USB 下载固件的功能，必须保证 USB 2.0 PHY 的三路供电均正常；
2. 系统上电后，所有 USB 2.0 PHY 默认处于 Normal mode，软件在 U-Boot SPL 阶段，配置 USB 2.0 PHY1/PHY2/PHY3 处于最低功耗 IDDQ mode（SDK 已经支持），在进入系统后，内核 USB 驱动会根据应用需求，设置对应的 USB 2.0 PHY 退出 IDDQ mode；
3. 对于内核已启用的 USB 2.0 PHY，内核 USB 2.0 PHY 驱动会自动对 PHY 进行动态功耗控制，当检测到有设备插入时，自动设置 USB 2.0 PHY 处于 Normal mode，当检测到没有设备插入时，自动设置 USB 2.0 PHY 处于 Suspend mode；

USB 2.0 PHY 处于不同工作模式的功耗数据如下表 4 所示。

表 4 USB 2.0 PHY 功耗数据 (统计为单个 USB 2.0 PHY 的功耗)

供电电源	读写数据	动态休眠	PHY disabled	二级待机	Unit
USB20_DVDD_0V75	8.9	2.8	0.05	0	mA
USB20_AVDD_1V8	8.6	3.34	0.05	0	mA
USB20_AVDD_3V3	2.5	0.14	0.05	0	mA

Note:

1. 读写数据功耗的测试场景：接 U2 盘拷贝数据，PHY 处于 Normal mode；
2. 动态休眠功耗的测试场景：USB 2.0 PHY 的 DTS enable，但不接 USB 外设，PHY 处于 Suspend mode；
3. PHY disabled 功耗的测试场景：USB 2.0 PHY 的 DTS diabled，PHY 处于 IDDQ mode；
4. 二级待机功耗的测试场景：USB 2.0 PHY 的三路供电电源全部关闭；

3.2.2 USB 3.1 PHY 供电及功耗管理

RK3576 支持两种 USB 3.1 Combo PHY：

1. USB 3.1/DP Combo PHY
2. USB 3.1/PCIe/SATA Combo PHY

这两种 USB 3.1 Combo PHY 对应的供电电源和功耗控制方式都不一样，下面分别进行说明。

3.2.2.1 USB 3.1/DP Combo PHY

RK3576 USB3.1 OTG0 使用 USB 3.1/DP Combo PHY 。在芯片内部，USB 3.1/DP Combo PHY 属于 VD_USBDPPHY (Alive)，在芯片外部有两路独立供电电源，如下图 6 所示。

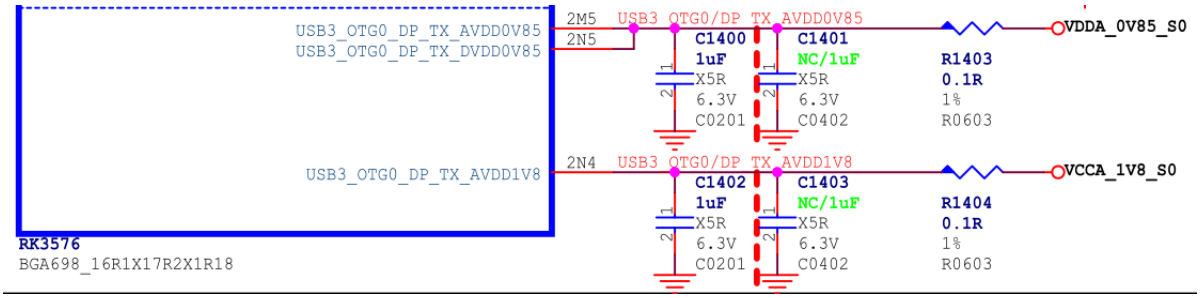

图 6 RK3576 USBDP Combo PHY 供电电源

表 5 USB 3.1/DP Combo PHY 供电电压要求

供电电源	最小	正常	最大	Unit
VDD_0V85/VDDA_0V85	0.8075	0.85	0.8925	V
VDDH_1V8	1.71	1.8	1.89	V

USB 3.1/DP Combo PHY 的功耗控制策略如下：

1. 系统上电后，USBDP PHY 处于未初始化状态时，功耗最低；
2. 在支持 USBDP 的应用场景，内核 USBDP PHY 驱动会自动对 PHY 进行动态功耗控制，当检测到有设备插入时，自动设置 USBDP PHY 处于 P0 State，当检测到没有设备插入时，自动设置 USBDP PHY 处于 P3 State (应用于 Type-A 接口)或者处于 reset state (应用于 Type-C 接口)；
3. 对于不使用的 USBDP 的应用场景（即 USB3.1 和 DP 都不使用），USBDP PHY 的供电电源可以根据项目需求选择正常供电或者断电两种电路设计，具体说明如下：

   (1) 如果要支持 USB3.1 下载固件的功能，则要求 USBDP PHY 的供电电源必须正常供电；

   (2) 如果不需要支持 USB3.1 下载固件的功能，则建议 USBDP PHY 的供电电源进行外部断电处理，但 DTS 要修改配置，具体请参考USB PHY 不供电的 DTS 配置；

   (3) 如果不需要支持 USB3.1 下载固件的功能，且 USBDP PHY 的供电电源正常供电，则建议将 USBDP PHY DTS 节点配置为 disabled，也即让 PHY 处于上电且未初始化状态，功耗最低；

表 6 USB 3.1/DP Combo PHY 功耗数据

供电电源	读写数据	动态休眠	PHY disabled	二级待机	Unit
VDD_0V85/VDDA_0V85	101.6	5	2	0	mA
VDDH_1V8	29	0	0	0	mA

Note：

1. 读写数据功耗的测试场景：接 U3 盘拷贝数据，PHY 处于 P0 state；
2. 动态休眠功耗的测试场景：Type-C 接口，不接 USB 外设，PHY 处于 reset state；
3. PHY disabled 功耗的测试场景：PHY 的 DTS 节点配置为 disabled，PHY 处于未初始化状态，此状态下，功耗最低；

4. 二级待机功耗的测试场景：PHY 的两路供电电源全部关闭；

内核 disable USBDP PHY 的方法如下：

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

&usbdp_phy_dp {
        status = "disabled";
};

&usbdp_phy_u3 {
        status = "disabled";
};
```

3.2.2.2 USB 3.1/PCIe/SATA Combo PHY

RK3576 USB3.1 OTG1使用 USB3.1/PCIe/SATA Combo PHY1。在芯片内部，这个 PHY 属于 PD_BUS (Alive)，在芯片外部有两路独立供电电源，如图 7 所示。

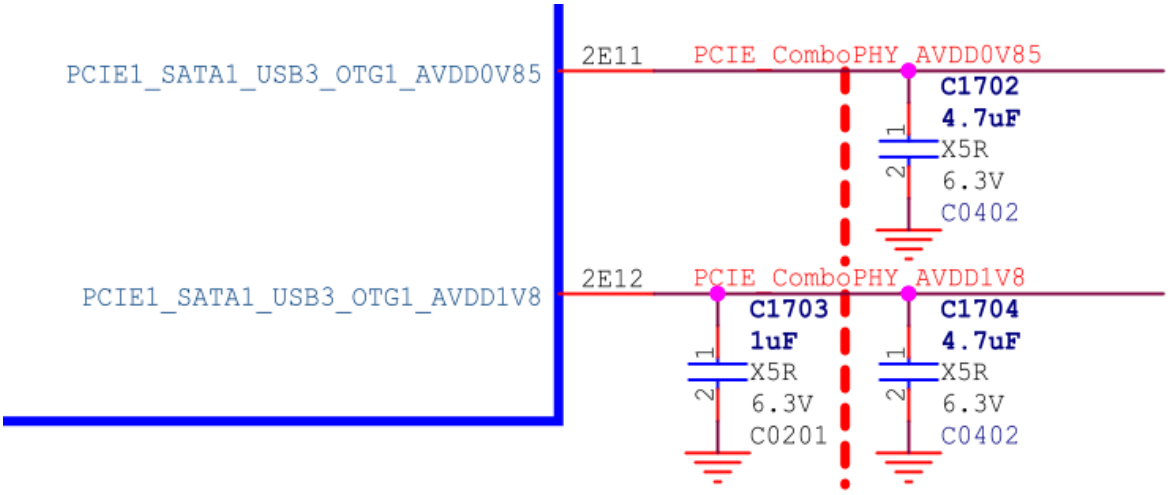

图 7 RK3576 USB 3.1/PCIe/SATA Combo PHY 供电电源

表 7 USB 3.1/PCIe/SATA Combo PHY 供电电压要求

供电电源	最小	正常	最大	Unit
AVDD_0V85	0.8	0.85	0.935	V
AVDD_1V8	1.62	1.8	1.98	V

USB 3.1/PCIe/SATA Combo PHY 的功耗控制策略如下：

1. 芯片上电时，USB 3.1/PCIe/SATA Combo PHY 默认处于工作状态。软件在 U-Boot SPL 阶段，设置 PHY 处于 reset state，以保持 PHY 处于最低功耗。进入内核后，USB 控制器驱动会通过调用 `rockchip_combphy_init()` 函数释放 PHY 的 reset。
2. PHY 的动态功耗控制：当 Combo PHY 工作在 USB mode 时，PHY 的 PIPE state (P0/P1/P2/P3) 由 USB 控制器硬件自动控制，根据不同的工作场景，动态进入和退出 P0/P1/P2/P3 state。比如，未插入任何 USB 设备时，PIPE 处于 P3 state；插入 U3 disk 时，则切换到 P0 state；当接 U3 HUB 时，只要 HUB 的下行端口没有接其他 USB 外设，则 PIPE state 会自动进入 P3 state 低功

耗。当有 USB 外设插入U3 HUB，则 PIPE state 自动切换为 P0。(注：P0 为正常工作状态，P3 为最低功耗状态)

3. 当明确不使用 USB 3.1 OTG1/PCIe/SATA 接口时，对应的 Combo PHY 可以不供电，但 DTS 要修改配置，具体请参考USB PHY 不供电的 DTS 配置；
4. 在 PHY 供电的情况下，如果不使用这个PHY，需要将对应的 PHY DTS 节点配置为 disabled，也即让 PHY 处于 reset state，功耗最低；

表 8 USB 3.1/SATA/PCIe Combo PHY 功耗数据

供电电源	读写数据	动态休眠	PHY disabled	二级待机	Unit
AVDD_0V85	44.5	9.6	0.4	0	mA
AVDD_1V8	5.2	0.5	0.2	0	mA

Note:

1. 读写数据功耗的测试场景：接 U3 盘拷贝数据，PHY 处于 P0 state；
2. 动态休眠的测试场景：PHY DTS enable，但不接 USB 外设，PHY 处于 P3 State；
3. PHY disabled 的测试场景：PHY 的 DTS 节点配置为 disabled，PHY 处于 reset state；
4. 二级待机功耗的测试场景：PHY 的两路供电电源全部关闭。

内核 disable USB 3.1/PCIe/SATA Combo PHY 的方法如下：

```
&combphy1_psu {
        status = "disabled";
};
```

3.3 USB 硬件电路设计

3.3.1 USB2_OTG0_VBUSDET 电路设计

USB2_OTG0_VBUSDET 用于 USB 作为 Device 时，设备的连接、断开检测。其设计注意点如下：

1. 设计为支持 PD 功能的 Type-C 接口，即支持外置 Type-C 控制器芯片（FUSB302 或 HUSB311 等），则参考 RK3576 EVB1 Type-C 的电路设计即可（ USB2_OTG0_VBUSDET 固定上拉到 VCC_3V3_S0），软件驱动可以通过 Type-C 控制器芯片的 CC 检测 USB Device 的连接和断开;
2. 对于其他没有支持外置 Type-C 控制器芯片的电路设计方案（如Type-C USB 2.0 only，Type-A USB 3.1，Micro USB 2.0/3.1），要求 USB2_OTG0_VBUSDET 仍然按照传统的分压电路设计，连接到 USB 接口的 VBUS 引脚，VBUSDET 不作常供电的设计（如果有作 USB HOST 的需求，需要独立的 GPIO 或者 PMIC VBUS 控制，不与其他 USB HOST VBUS控制电路复用）；
3. 要求芯片输入端 USB2_OTG0_VBUSDET 的高电平范围在 **[0.9V ~ 3.3V]**;

3.3.2 Maskrom USB 电路设计

Maskrom 支持 USB3.1 下载固件，同时向下兼容 USB2.0 下载固件。硬件设计要求如下：

1. USB 2.0 PHY 相关电源必须供电，具体请参考USB 2.0 PHY 供电及功耗管理；
2. USBDP PHY 相关电源，可以根据实际项目需求选择正常供电或者断电两种电路设计，具体请参考USB 3.1 PHY 供电及功耗管理；

   如果要支持 USB3.1 下载固件，要求 USBDP PHY 相关电源必须供电，并且 USB3_OTG0_TX1/RX1 连接到 USB 下载口。
3. Maskrom USB 枚举不依赖于 USB2_OTG0_VBUSDET 的电平；
4. 烧写工具配置说明

   (1) USB 3.1 下载固件的功能，必须配合 Windows 工具 v3.28 或者 Linux 工具 v2.30 或者更新的版本使用，并手动修改烧写工具的 config.ini

   Windows 工具的 config.ini 修改：USB3_TRANSFER=TRUE

   Linux 工具的 config.ini 修改：usb3_transfer_on=true

   (2) 硬件电路设计上，如果只支持 OTG0 USB2 only 且 USBDP PHY 未供电 (即 USB3_OTG0_DP_TX_AVDD0V85、USB3_OTG0_DP_TX_DVDD0V85、USB3_OTG0_DP_TX_AVDD1V8 这三路未供电)，则需要关闭烧写工具的 USB3 功能，否则会导致 Maskrom下载固件失败。

3.3.3 USB2 支持唤醒系统的电路设计

USB2 支持通过 DP/DM 唤醒系统，该功能对二级待机时的供电有特殊要求，具体如下：

1. USB2 PHY 的三路外部电源（USB2_OTG_DVDD_0V75、USB2_OTG_AVDD_1V8、USB2_OTG_AVDD_3V3）要保持供电；
2. USB 接口的 VBUS 保持对 USB 外设供电；
3. RK3576 的 OSC 外部电源（OSC_AVDD1V8）保持供电；
4. RK3576 的 PMU 外部电源保持供电；
5. RK3576 的 Logic 外部电源可以选择供电或者断电。建议 Logic 断电，以降低二级待机功耗。

3.3.4 Type-C USB 3.1/DP 全功能硬件电路

以 RK3576 EVB1 USB 3.1 OTG0 Type-C 硬件电路设计为例。

1. Type-C 电路须配合外置的 Type-C 控制器芯片，才能实现 Type-C 的完整功能（包括：正反面检测、PD充电协商、Alternate Mode 协商）。RK3576 可支持大部分常用的 Type-C 控制器芯片，具体请参考Type-C 控制器芯片支持列表；
2. 为了支持高压充电功能，同时降低硬件电路的风险，USB2_OTG0_VBUSDET 不要连接 Type-C 接口的 VBUS，固定上拉到 3.3V 即可（如图 8 和图 9 USB2_OTG0_VBUSDET 连接到 VCC_3V3_S0），但不能悬空；
3. USB2_OTG0_ID 只用于 Micro 接口类型的 OTG 功能，Type-C 电路不需要使用，悬空即可。

4. TYPEC_SBU1/TYPEC0_SBU2 只用于 DP Alternate Mode 的 AUX 通信。按照 AUX 的协议要求，需要根据 Type-C 插入的正反面，对 SBU1/SBU2 进行相应的电平上拉操作。 因为 RK3576 芯片内部没有实现 SBU1/SBU2 的自动上拉，所以要求硬件外部电路增加两个 GPIO 控制（对应图 9 中的 TYPEC_SBU1/TYPEC_SBU2）。对 GPIO 的默认上下拉方式没要求，可以选择任意的 GPIO。软件上，需要修改 usbdp_phy 节点的属性 `sbu1-dc-gpios` 和 `sbu2-dc-gpios` 进行适配；
5. Type-C USB DTS 的软件配置，请参考 Type-C USB 3.1/DP 全功能 DTS 配置；

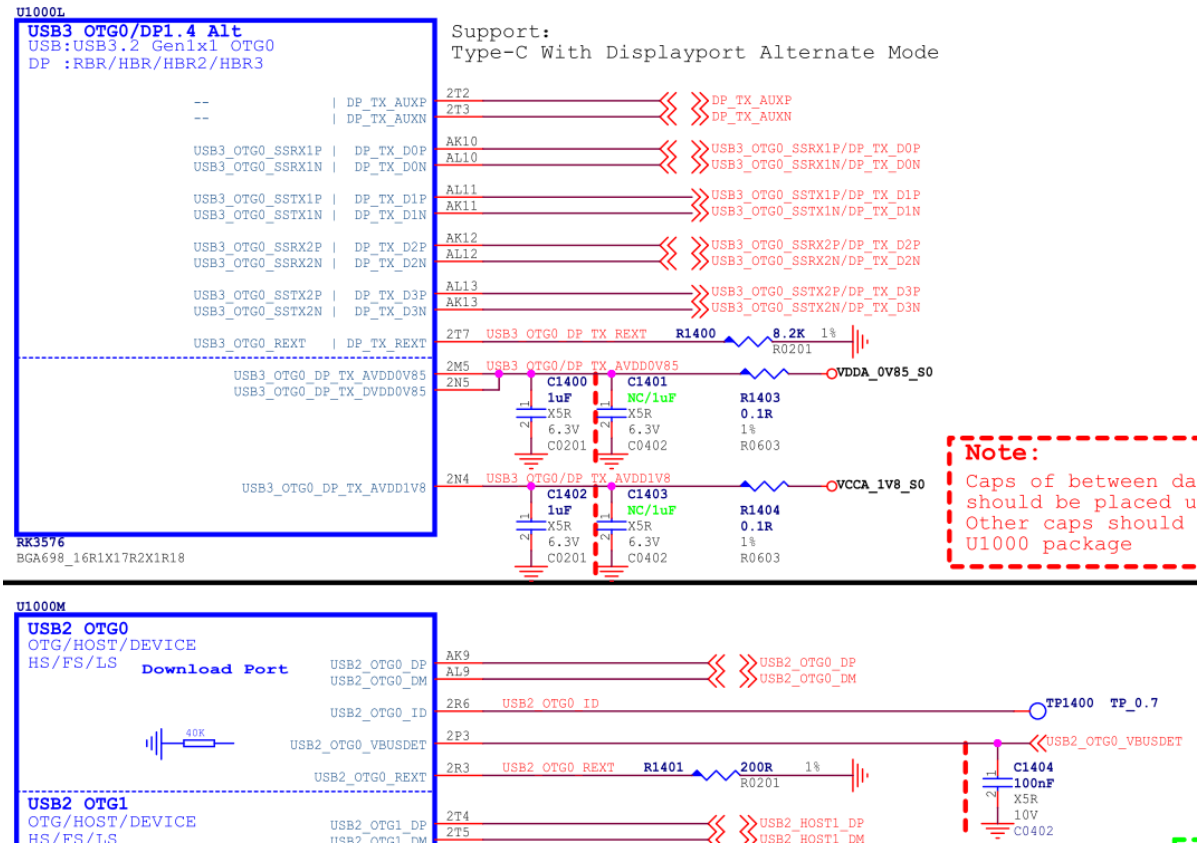

图 8 RK3576 Type-C 电路

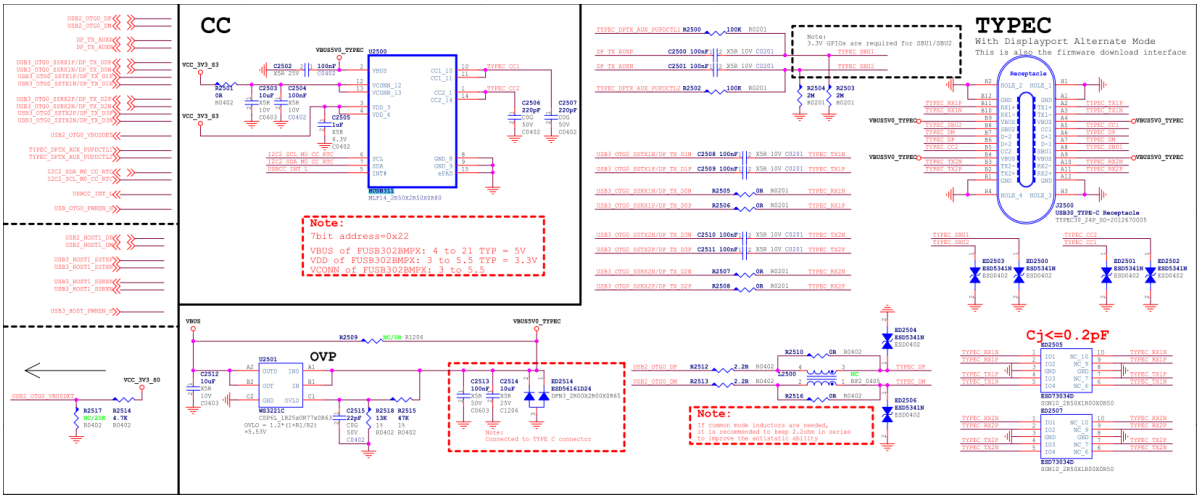

图 9 RK3576 Type-C 接口

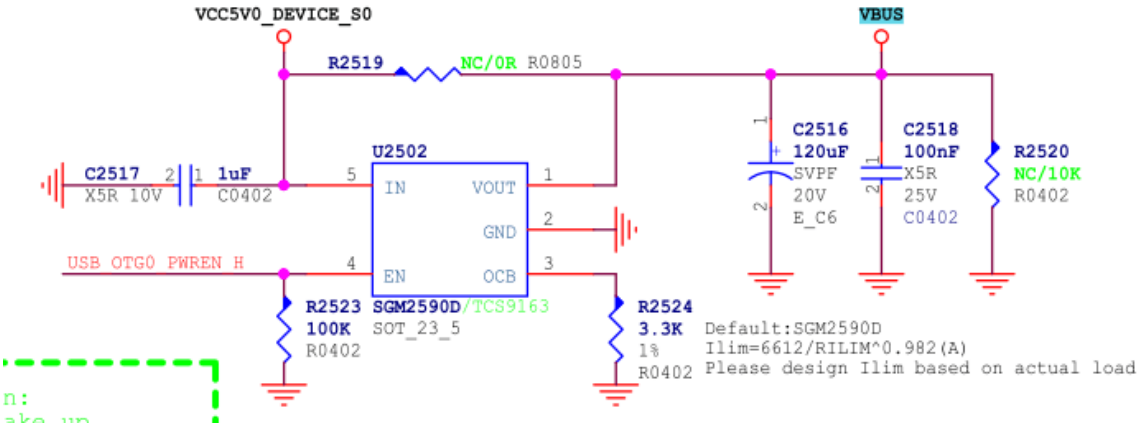

图 10 RK3576 Type-C VBUS 控制电路

3.3.5 Type-C to Type-A USB 3.1/DP 硬件电路

该方案适用于 RK3576 USB 3.1 OTG0 Type-C 可以拆分为独立的 Type-A USB 3.1 接口和 DP（2 x Lane）接口使用。以 RK3576 EVB2 Type-C to Type-A USB 3.1/DP 硬件电路设计为例。

1. Type-A USB 3.1 使用 USB3_OTG0_SSRX1P/N 和 USB3_OTG0_SSTX1P/N (对应芯片内部 USBDP PHY 的 lane0/1)，而 DP 使用 DP_TX_D2P/N 和 DP_TX_D3P/N (对应芯片内部 USBDP PHY 的 lane2/3)；
2. 如果 USB OTG 有作 Device/HOST 的应用场景，建议 USB2_OTG0_VBUSDET 通过 30KΩ 的电阻串联到 Type-A USB 接口的 VBUS；
3. Type-A VBUS 的供电电源 (VCC5V0_USB30_HOST0) 由 GPIO 控制，当 OTG 作 Device mode，关闭 VBUS 输出。当 OTG 作 HOST mode，打开 VBUS 输出。此外，稳压芯片 SGM2590D 的输出电流由 OCB pin 连接的电阻决定，最大电流 Ilim(A)=6612/RILIM^0.982 ，如下图 12 所示，VBUS 输出限流为 2.3A；
4. Type-C to Type-A USB 3.1/DP 对应的 DTS 配置，请参考 Type-C to Type-A USB 3.1/DP DTS 配置；

Note：

理论上，硬件电路也可以设计为 Type-A USB 3.1 使用 lane2/3，DP 使用 lane0/1，但软件需要对 usbdp_phy 节点的属性 `rockchip,dp-lane-mux` 进行修改，以适配硬件设计。

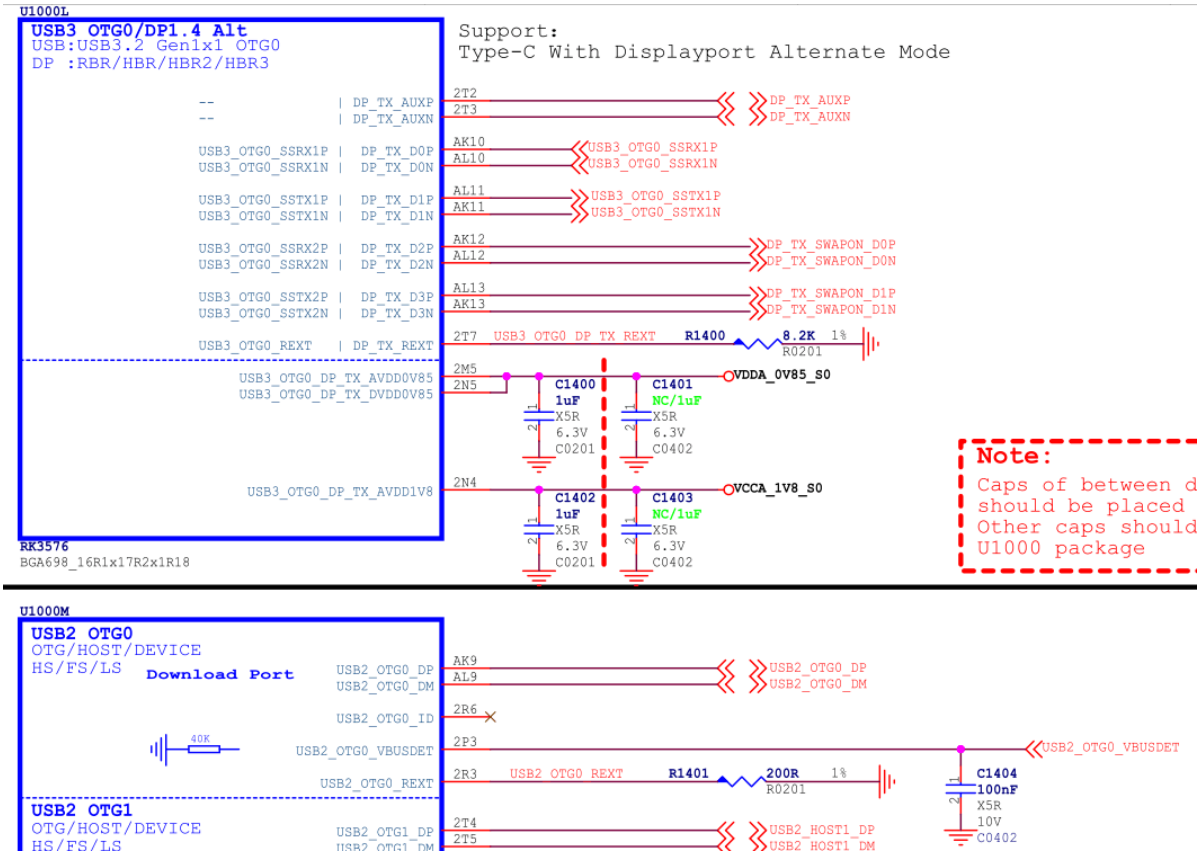

图 11 RK3576 Type-A USB 3.1/DP 电路

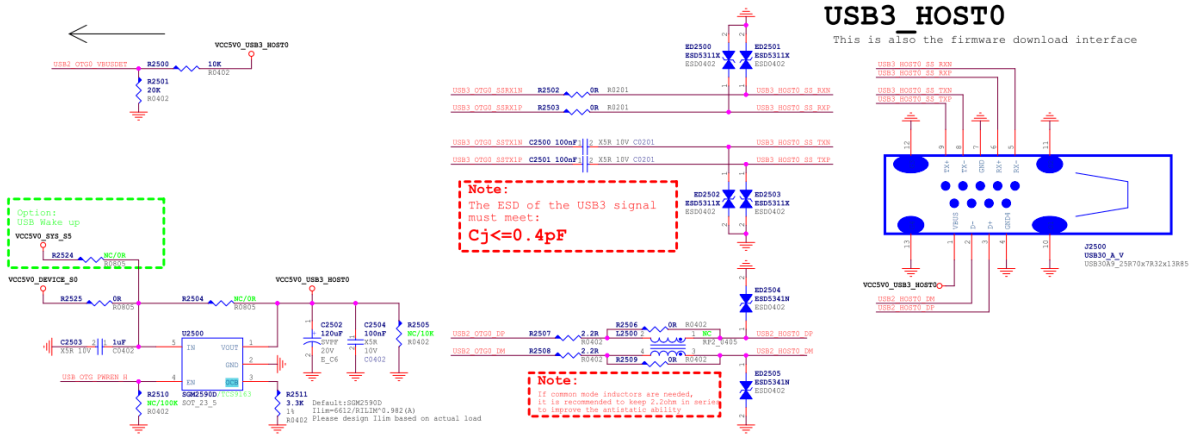

图 12 RK3576 Type-A USB3.1 接口

3.3.6 Type-C to Type-A/Micro USB 2.0/DP 硬件电路

该方案适用于 RK3576 USB 3.1 OTG0 Type-C可以拆分为独立的 Type-A USB 2.0 接口和 DP（4 x Lane）接口使用。以 RK3576 TEST2 Board Type-C to Type-A USB 2.0/DP 硬件电路设计为例。

1. TYPEC USBDP PHY 的 4 x Lane 全部给 DP 接口使用，USB 不使用 USBDP PHY；
2. TYPEC USB 只作 HOST mode 使用时，USB2_OTG0_ID 和 USB2_OTG0_VBUSDET 悬空即可；
3. Type-A VBUS 的供电电源 (VCC5V0_USB2_OTG0) 由 GPIO 控制，当 OTG 作 HOST mode，打开 VBUS 输出。此外，稳压芯片 SGM2590D 同样有限流配置；
4. Type-C to Type-A USB 2.0/DP 对应的 DTS 配置，请参考 Type-C to Type-A/Micro USB 2.0/DP DTS 配置；

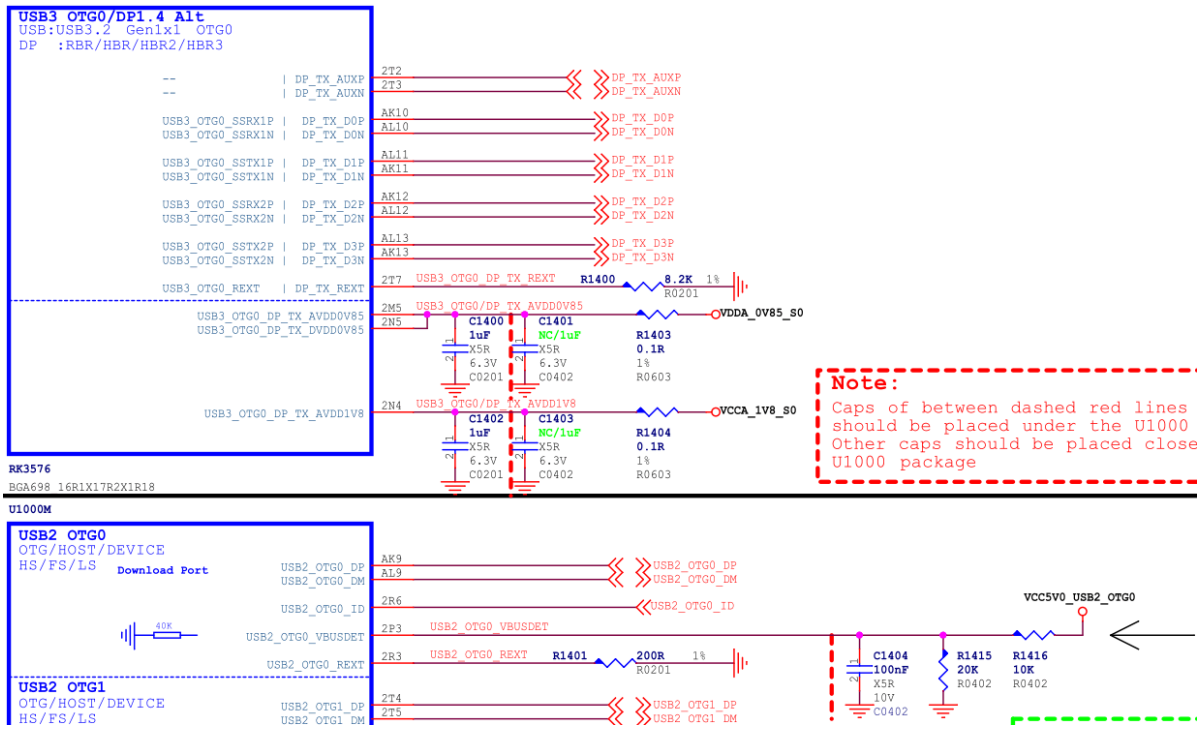

图 13 RK3576 Type-A USB 2.0/DP 电路

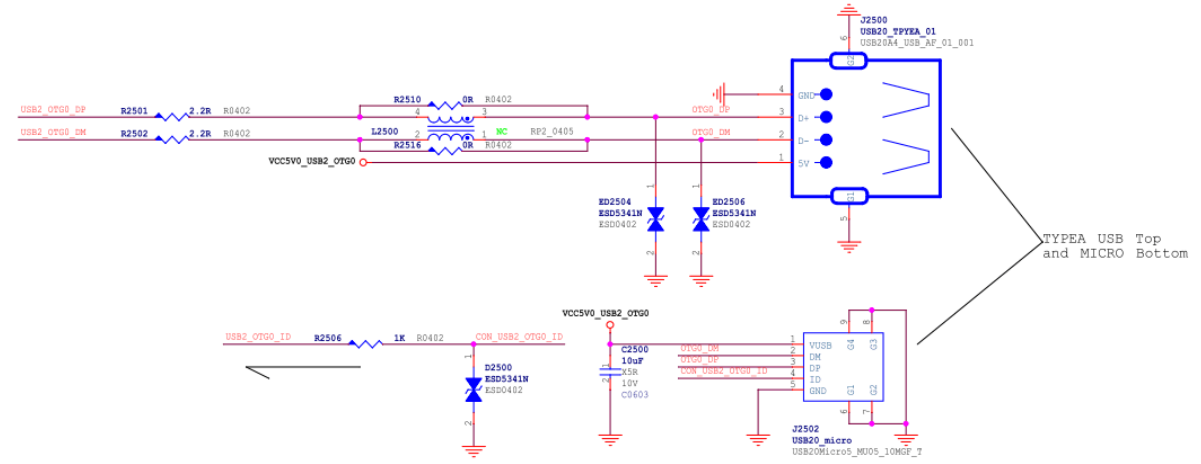

图 14 RK3576 Type-A/Micro USB2 接口

3.3.7 Type-A USB3.1 OTG1 硬件电路

如前文所述，RK3576 USB3.1 OTG0 和 OTG1 的关键区别在于连接的 USB3 PHY 不同，因此，R3576 USB 3.1 OTG1 非 Type-C 硬件电路基本和 USB 3.1 OTG0 USB 电路相同；而对于全功能 Type-C 电路，需要在 OTG1 和 Type-C 接口中间增加一个 USB3.1 Switch 芯片（如 FUSB340）和一个 Type-C 控制器（如 FUSB302）。

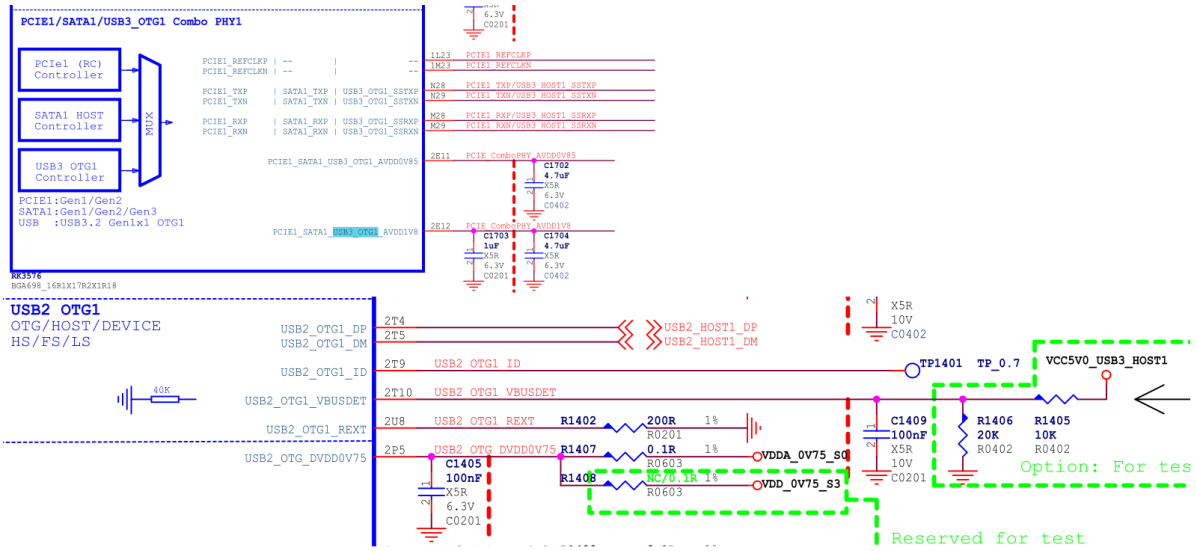

图 15 RK3576 USB3.1 OTG1 接口

4. RK3576 USB DTS 配置

RK3576 USB DTS 配置，包括：芯片级 USB 控制器/PHY DTSI 配置和板级 DTS 配置。详细配置方法，请参考内核如下文档：

1. kernel/Documentation/devicetree/bindings/usb/snps,dwc3.yaml
2. kernel/Documentation/devicetree/bindings/connector/usb-connector.yaml
3. kernel/Documentation/devicetree/bindings/phy/phy-rockchip-inno-usb2.yaml
4. kernel/Documentation/devicetree/bindings/phy/phy-rockchip-usbdp.yaml
5. kernel/Documentation/devicetree/bindings/phy/phy/phy-rockchip-naneng-combphy.txt

4.1 USB 芯片级 DTSI 配置

RK3576 DTSI 文件中 USB 控制器和 PHY 相关的主要节点如下表所示，由于 USB DTSI 节点配置的是 USB 控制器和 PHY 的公共资源和属性，建议开发者不要改动。对应的 DTSI 完整路径如下：

`arch/arm64/boot/dts/rockchip/RK3576.dtsi`

表 9 RK3576 USB 接口和 USB DTS 节点的对应关系

USB 接口名称(原理图)	USB 控制器 DTS 节点	USB PHY DTS 节点
USB3 OTG0	usb_drd0_dwc3	u2phy0 u2phy0_otg usbdp_phy usbdp_phy_u3
USB3 OTG1	usb_drd1_dwc3	u2phy1 u2phy1_otg combphy1_psu

USB 控制器 DTSI 节点如下：

```
#USB3.1 OTG0 Controller
usb_drd0_dwc3: usb@23000000 {
        compatible = "rockchip,rk3576-dwc3", "snps,dwc3";
        ......
};

#USB3.1 OTG1 Controller
usb_drd1_dwc3: usb@23400000 {
        compatible = "rockchip,rk3576-dwc3", "snps,dwc3";
        ......
};
```

USB PHY DTSI 节点如下：

注意：USB PHY 和 USB 控制器具有一一对应的关系，需要成对配置。在芯片内部，USB PHY 和 控制器的连接关系，请参考 RK3576 USB 控制器和 PHY 简介的图 1 和表 9。在DTSI 节点中，通过 USB 控制器节点的 "phys" 属性关联对应的 USB PHY。

```
usb2phy_grf: syscon@2602e000 {
        compatible = "rockchip,RK3576-usb2phy-grf", "syscon",
                     "simple-mfd";
        ......

        // USB2.0 PHY0
        u2phy0: usb2-phy@0 {
                compatible = "rockchip,RK3576-usb2phy";
                ......
                u2phy0_otg: otg-port {
                        #phy-cells = <0>;
                        status = "disabled";
                };
        };

        // USB2.0 PHY1
        u2phy1: usb2-phy@2000 {
                compatible = "rockchip,RK3576-usb2phy";
                ......
                u2phy1_otg: otg-port {
                        #phy-cells = <0>;
                        status = "disabled";
                };
        };
};

#USB3.1/DP Combo PHY
usbdp_phy: phy@2b010000 {
        compatible = "rockchip,rk3576-usbdp-phy";
        ......
        usbdp_phy_dp: dp-port {
                #phy-cells = <0>;
                status = "disabled";
        };
```

```
        usbdp_phy_u3: u3-port {
                #phy-cells = <0>;
                status = "disabled";
        };
};

#USB3.1/SATA/PCIe PHY1
combphy1_psu: phy@2b060000 {
        compatible = "rockchip,rk3576-naneng-combphy";
        ......
};
```

4.2 Type-C USB 3.1/DP 全功能 DTS 配置

参考 `arch/arm64/boot/dts/rockchip/rk3576-evb1.dtsi` 中 usbc0 接口的 DTS 配置。

```
#USB2.0 PHY配置属性"rockchip,typec-vbus-det"，表示支持Type-C VBUS_DET常拉高的硬件设计
&u2phy0_otg {
        rockchip,typec-vbus-det;
};

#USB3.1/DP PHY，需要根据硬件设计，配置属性"sbu1-dc-gpios"和"sbu2-dc-gpios"
&usbdp_phy {
        orientation-switch;
        svid = <0xff01>;
        sbu1-dc-gpios = <&gpio2 RK_PA6 GPIO_ACTIVE_HIGH>;
        sbu2-dc-gpios = <&gpio2 RK_PA7 GPIO_ACTIVE_HIGH>;

        port {
                #address-cells = <1>;
                #size-cells = <0>;
                usbdp_phy_orientation_switch: endpoint@0 {
                        reg = <0>;
                        remote-endpoint = <&usbc0_orien_sw>;
                };

                usbdp_phy_dp_altmode_mux: endpoint@1 {
                        reg = <1>;
                        remote-endpoint = <&dp_altmode_mux>;
                };
        };
};

#USB3.1 OTG0 Controller
&usb_drd0_dwc3 {
        dr_mode = "otg";
        usb-role-switch;
        port {
                usb_drd0_role_switch: endpoint {
                        remote-endpoint = <&usbc0_role_sw>;
                };
        };
};
```

```
#VBUS GPIO配置，在Type-C控制器芯片驱动中控制该GPIO
vbus5v0_typec: vbus5v0-typec {
        compatible = "regulator-fixed";
        regulator-name = "vbus5v0_typec";
        regulator-min-microvolt = <5000000>;
        regulator-max-microvolt = <5000000>;
        enable-active-high;
        gpio = <&gpio0 RK_PD1 GPIO_ACTIVE_HIGH>;
        vin-supply = <&vcc5v0_device>;
        pinctrl-names = "default";
        pinctrl-0 = <&usb_otg0_pwren>;
};

#配置外置Type-C控制器芯片HUSB311
#需要根据实际的硬件设计，配置"I2C/interrupts/vbus-supply/usb_con"的属性
&i2c2 {
        status = "okay";

        usbc0: husb311@4e {
                compatible = "hynetek,husb311"; // 注意本 Note 1
                reg = <0x4e>;
                interrupt-parent = <&gpio0>;
                interrupts = <RK_PA5 IRQ_TYPE_LEVEL_LOW>;
                pinctrl-names = "default";
                pinctrl-0 = <&usbc0_int>;
                vbus-supply = <&vbus5v0_typec>;
                status = "okay";

                port {
                        usbc0_role_sw: endpoint {
                                remote-endpoint = <&usb_drd0_role_switch>;
                        };
                };

                usb_con: connector {
                        compatible = "usb-c-connector";
                        label = "USB-C";
                        data-role = "dual";
                        power-role = "dual";
                        try-power-role = "sink";
                        op-sink-microwatt = <1000000>;

                        // 须根据实际的硬件设计进行配置充电/供电能力
                        sink-pdos =
                                <PDO_FIXED(5000, 1000, PDO_FIXED_USB_COMM)>;
                        source-pdos =
                                <PDO_FIXED(5000, 3000, PDO_FIXED_USB_COMM)>;

                        altmodes {
                                #address-cells = <1>;
                                #size-cells = <0>;

                                altmode@0 {
                                        reg = <0>;
                                        svid = <0xff01>;
```

```
                                        vdo = <0xffffffff>;
                                };
                        };

                        ports {
                                #address-cells = <1>;
                                #size-cells = <0>;

                                port@0 {
                                        reg = <0>;
                                        usbc0_orien_sw: endpoint {
                                                remote-endpoint =
<&usbdp_phy_orientation_switch>;
                                        };
                                };

                                port@1 {
                                        reg = <1>;
                                        dp_altmode_mux: endpoint {
                                                remote-endpoint =
<&usbdp_phy_dp_altmode_mux>;
                                        };
                                };
                        };
                };
        };
```

Note：使用 FUSB302/AW35615/AW35615 芯片替代 HUSB311 芯片的软件修改

1. DTS 修改：只需要基于 HUSB311 的 DTS 配置进行简单修改即可，参考修改如下：

```
#配置外置Type-C控制器芯片HUSB311
&i2c2 {
        usbc0: fusb302@22 {
                compatible = "fcs,fusb302";
                reg = <0x22>;
                ..... // 其它同HUSB311
        };
};
```

2. 驱动软件修改：修改 FUSB302 驱动的 irq wakeup 机制，以支持待机 VBUS 断电或者供电等各种应用场景。

```
diff --git a/drivers/usb/typec/tcpm/fusb302.c b/drivers/usb/typec/tcpm/fusb302.c
index 77a050cd2e88..72bef94887b4 100644
--- a/drivers/usb/typec/tcpm/fusb302.c
+++ b/drivers/usb/typec/tcpm/fusb302.c
@@ -105,6 +105,7 @@ struct fusb302_chip {
        bool vbus_on;
        bool charge_on;
        bool vbus_present;
+       bool wakeup;
        enum typec_cc_polarity cc_polarity;
        enum typec_cc_status cc1;
        enum typec_cc_status cc2;
```

```
@@ -1769,7 +1770,8 @@ static int fusb302_probe(struct i2c_client *client,
                dev_err(dev, "cannot request IRQ for GPIO Int_N, ret=%d", ret);
                goto tcpm_unregister_port;
        }
-       enable_irq_wake(chip->gpio_int_n_irq);
+       chip->wakeup = device_property_read_bool(dev, "wakeup-source");
+       device_init_wakeup(dev, true);
        i2c_set_clientdata(client, chip);

        return ret;
@@ -1788,7 +1790,7 @@ static void fusb302_remove(struct i2c_client *client)
 {
        struct fusb302_chip *chip = i2c_get_clientdata(client);

-       disable_irq_wake(chip->gpio_int_n_irq);
+       device_init_wakeup(chip->dev, false);
        free_irq(chip->gpio_int_n_irq, chip);
        kthread_destroy_worker(chip->irq_worker);
        cancel_delayed_work_sync(&chip->bc_lvl_handler);
@@ -1809,6 +1811,11 @@ static int fusb302_pm_suspend(struct device *dev)

        /* Make sure any pending irq_work is finished before the bus suspends */
        kthread_flush_worker(chip->irq_worker);
+
+       if (device_may_wakeup(dev) && (!chip->vbus_on || chip->wakeup))
+               enable_irq_wake(chip->gpio_int_n_irq);
+       else
+               disable_irq(chip->gpio_int_n_irq);
        return 0;
 }

@@ -1819,6 +1826,11 @@ static int fusb302_pm_resume(struct device *dev)
        u8 pwr;
        int ret = 0;

+       if (device_may_wakeup(dev) && (!chip->vbus_on || chip->wakeup))
+               disable_irq_wake(chip->gpio_int_n_irq);
+       else
+               enable_irq(chip->gpio_int_n_irq);
+
        /*
         * When the power of fusb302 is lost or i2c read failed in PM S/R
         * process, we must reset the tcpm port first to ensure the devices
```

4.3 Type-C to Type-A USB 3.1/DP DTS 配置

参考 `arch/arm64/boot/dts/rockchip/RK3576-evb2.dtsi` Type-C to Type-A USB 3.1/DP 的 DTS 配置。

```
#USB2.0 PHY配置"phy-supply"属性，用于控制VBUS输出5V
&u2phy0_otg {
        phy-supply = <&vcc5v0_otg>;
};
```

```
#VBUS GPIO配置，在USB2.0 PHY驱动中控制该GPIO
vcc5v0_otg: vcc5v0-otg {
        compatible = "regulator-fixed";
        regulator-name = "vcc5v0_otg";
        regulator-min-microvolt = <5000000>;
        regulator-max-microvolt = <5000000>;
        enable-active-high;
        gpio = <&gpio0 RK_PD1 GPIO_ACTIVE_HIGH>;
        vin-supply = <&vcc5v0_device>;
        pinctrl-names = "default";
        pinctrl-0 = <&usb_otg0_pwren>;
};

#USB3.1/DP PHY0，只需配置DP使用lane2/3,驱动会自动分配lane0/1给USB3.1 Rx/Tx
#如果硬件设计DP使用lane0/1，则此处应配置"rockchip,dp-lane-mux = <0 1>"
#注意：实际电路中，即使未支持DP，也需要配置"rockchip,dp-lane-mux"，否则USBDP PHY驱动无法自动分配lane给USB3.1
&usbdp_phy {
        rockchip,dp-lane-mux = <2 3>;
};

#USB3.1 OTG0 Controller
#配置"dr_mode"为"otg"，同时配置"extcon"属性，才能支持软件切换Device/Host mode
&usb_drd0_dwc3 {
        dr_mode = "otg";
        extcon = <&u2phy0>;
        status = "okay";
};
```

4.4 Type-C to Type-A/Micro USB 2.0/DP DTS 配置

参考 `arch/arm64/boot/dts/rockchip/RK3576-test2.dtsi` Type-C to Type-A USB 2.0/DP 的 DTS 配置。

```
#USB2.0 PHY0配置
#配置属性"phy-supply"，用于控制VBUS输出5V
#配置属性"rockchip,sel-pipe-phystatus"，表示选择GRF控制pipe phystatus，替代USBDP PHY的控制
#配置属性"rockchip,dis-u2-susphy"，otg mode为可选项，表示关闭USB2 PHY驱动动态进入suspend mode的功能
&u2phy0_otg {
        vbus-supply = <&vcc5v0_otg>;
        rockchip,sel-pipe-phystatus;
        rockchip,dis-u2-susphy;
        status = "okay";
};

#VBUS GPIO配置，在USB2.0 PHY驱动中控制该GPIO
vcc5v0_otg: vcc5v0-otg {
        ...
};
```

```
#USB3.1/DP PHY，配置DP使用lane0/1/2/3
#需要根据实际的硬件设计，配置属性"rockchip,dp-lane-mux"
#配置属性maximum-speed = "high-speed"，通知USBDP PHY驱动将USB限制为USB2.0 only
&usbdp_phy {
        maximum-speed = "high-speed";
        rockchip,dp-lane-mux = < 0 1 2 3 >;
        status = "okay";
};

&usbdp_phy_dp {
        status = "okay";
};

&usbdp_phy_u3 {
        status = "okay";
};

#配置属性"maximum-speed"，通知DWC3驱动将USB限制为USB2.0 only
#配置属性"snps,dis_u2_susphy_quirk"，关闭控制器硬件suspend usb2 phy的功能，提高USB通信的稳定性
#配置属性"snps,usb2-lpm-disable"，关闭控制器Host mode的LPM功能，提高USB外设的兼容性
&usb_drd0_dwc3 {
        phys = <&u2phy0_otg>;
        phy-names = "usb2-phy";
        extcon = <&u2phy0>;
        maximum-speed = "high-speed";
        snps,dis_u2_susphy_quirk;
        snps,usb2-lpm-disable;
        status = "okay";
};
```

4.5 Type-C USB 2.0 only DTS 配置

配置1. 硬件电路集成外置 Type-C 控制器芯片，支持 PD

```
#USB2.0 PHY0注册typec orientation switch，用于与TCPM子系统交互，获取USB热拔插的信息
&u2phy0 {
        orientation-switch;
        status = "okay";

        port {
                u2phy0_orientation_switch: endpoint {
                        remote-endpoint = <&usbc0_orien_sw>;
                };
        };
};

#USB2.0 PHY0 OTG配置
#配置属性"rockchip,sel-pipe-phystatus",表示选择GRF控制pipe phystatus，替代USBDP PHY的控制
#配置属性"rockchip,typec-vbus-det"，表示支持Type-C VBUS_DET常拉高的硬件设计
#配置属性"rockchip,dis-u2-susphy"，可选项，表示关闭USB2 PHY驱动动态进入suspend mode的功能
&u2phy0_otg {
```

```
        rockchip,sel-pipe-phystatus;
        rockchip,typec-vbus-det;
        rockchip,dis-u2-susphy;
        status = "okay";
};

#disable USBDP PHY0的所有相关节点，让USBDP PHY0处于未初始化状态，达到最低功耗的目的
&usbdp_phy {
        status = "disabled";
};

&usbdp_phy_dp {
        status = "disabled";
};

&usbdp_phy_u3 {
        status = "disabled";
};

&dp {
        status = "disabled";
};

#配置USB3.1 OTG0 Controller
#配置"phys = <&u2phy0_otg>",即不引用USBDP PHY
#配置maximum-speed = "high-speed"，通知DWC3驱动将USB限制为USB2.0 only
#配置属性"snps,dis_u2_susphy_quirk"，关闭控制器硬件suspend usb2 phy的功能，提高USB通信的
稳定性
#配置属性"snps,usb2-lpm-disable"，关闭控制器Host mode的LPM功能，提高USB外设的兼容性
&usb_drd0_dwc3 {
        dr_mode = "otg";
        status = "okay";
        maximum-speed = "high-speed";
        phys = <&u2phy0_otg>;
        phy-names = "usb2-phy";
        usb-role-switch;
        snps,dis_u2_susphy_quirk;
        snps,usb2-lpm-disable;
        port {
                usb_drd0_role_switch: endpoint {
                        remote-endpoint = <&usbc0_role_sw>;
                };
        };
};

#配置外置Type-C控制器芯片HUSB311
#需要根据实际的硬件设计，配置"I2C/interrupts/vbus-supply/usb_con"的属性
#需要配置usbc0_orien_sw的属性remote-endpoint = <&u2phy0_orientation_switch>
&i2c2 {
        status = "okay";
        usbc0: husb311@4e {
                compatible = "hynetek,husb311";
                reg = <0x4e>;
                interrupt-parent = <&gpio0>;
                interrupts = <RK_PC4 IRQ_TYPE_LEVEL_LOW>;
                pinctrl-names = "default";
```

```
                pinctrl-0 = <&usbc0_int>;
                vbus-supply = <&vbus5v0_typec>;
                status = "okay";

                port {
                        usbc0_role_sw: endpoint {
                                remote-endpoint = <&usb_drd0_role_switch>;
                        };
                };

                usb_con: connector {
                        compatible = "usb-c-connector";
                        label = "USB-C";
                        data-role = "dual";
                        power-role = "dual";
                        ......
                        port {
                                usbc0_orien_sw: endpoint {
                                        remote-endpoint =
<&u2phy0_orientation_switch>;
                                };
                        };
                };
        };
```

配置2. 硬件电路不带外置 Type-C 控制器芯片，支持 Device only

Type-C USB 2.0 Device 的 DTS 配置

```
#USB2.0 PHY0 OTG配置
#配置属性"rockchip,sel-pipe-phystatus",表示选择GRF控制pipe phystatus，替代USBDP PHY的
控制
#配置属性"rockchip,dis-u2-susphy"，必选项，表示关闭USB2 PHY驱动动态进入suspend mode的功能
&u2phy0_otg {
        rockchip,sel-pipe-phystatus;
        rockchip,dis-u2-susphy;
        status = "okay";
};

#disable USBDP PHY的所有相关节点，让USBDP PHY0处于未初始化状态，达到最低功耗的目的
&usbdp_phy {
        status = "disabled";
};

&usbdp_phy_dp {
        status = "disabled";
};

&usbdp_phy_u3 {
        status = "disabled";
}

#配置USB3.1 OTG0 Controller
#配置dr_mode = "peripheral"，通知DWC3驱动初始化为Device only mode
#配置"phys = <&u2phy0_otg>",即不引用USBDP PHY
```

```
#配置maximum-speed = "high-speed"，通知DWC3驱动将USB限制为USB2.0 only
#配置属性"snps,dis_u2_susphy_quirk"，关闭控制器硬件suspend usb2 phy的功能，提高USB通信的稳定性
&usb_drd0_dwc3 {
        dr_mode = "peripheral";
        phys = <&u2phy0_otg>;
        phy-names = "usb2-phy";
        maximum-speed = "high-speed";
        snps,dis_u2_susphy_quirk;
        status = "okay";
};
```

配置3. 硬件电路不带外置 Type-C 控制器芯片，支持 OTG（**需要增加 CC to ID 电平转换电路**）

```
#USB2.0 PHY0 OTG配置
#配置属性"rockchip,sel-pipe-phystatus",表示选择GRF控制pipe phystatus，替代USBDP PHY的控制
#配置属性"rockchip,dis-u2-susphy"，otg mode为可选项，表示关闭USB2 PHY驱动动态进入suspend mode的功能
&u2phy0_otg {
        rockchip,sel-pipe-phystatus;
        rockchip,dis-u2-susphy;
        status = "okay";
};

#disable USBDP PHY的所有相关节点，让USBDP PHY0处于未初始化状态，达到最低功耗的目的
&usbdp_phy {
        status = "disabled";
};

&usbdp_phy_dp {
        status = "disabled";
};

&usbdp_phy_u3 {
        status = "disabled";
}

#配置USB3.1 OTG0 Controller
#配置dr_mode = "otg"
#配置"phys = <&u2phy0_otg>",即不引用USBDP PHY
#配置maximum-speed = "high-speed"，通知DWC3驱动将USB限制为USB2.0 only
#配置"extcon"属性，才能支持自动切换Device/Host mode
#配置属性"snps,dis_u2_susphy_quirk"，关闭控制器硬件suspend usb2 phy的功能，提高USB通信的稳定性
#配置属性"snps,usb2-lpm-disable"，关闭控制器Host mode的LPM功能，提高USB外设的兼容性
&usb_drd0_dwc3 {
        dr_mode = "otg";
        phys = <&u2phy0_otg>;
        phy-names = "usb2-phy";
        maximum-speed = "high-speed";
        extcon = <&u2phy0>;
        snps,dis_u2_susphy_quirk;
        snps,usb2-lpm-disable;
        status = "okay";
```

```
};
```

4.6 Type-A USB 3.1 DTS 配置

参考 `arch/arm64/boot/dts/rockchip/rk3576-evb1.dtsi` USB3 OTG1 的 DTS 配置

```
#USB2.0 PHY1配置"phy-supply"属性，用于控制VBUS输出5V
&u2phy1_otg {
        phy-supply = <&vcc5v0_host>;
}

#VBUS GPIO配置，在USB2.0 PHY驱动中控制该GPIO
vcc5v0_host: vcc5v0-host {
        compatible = "regulator-fixed";
        regulator-name = "vcc5v0_host";
        regulator-boot-on;
        regulator-always-on;
        regulator-min-microvolt = <5000000>;
        regulator-max-microvolt = <5000000>;
        enable-active-high;
        gpio = <&gpio0 RK_PC7 GPIO_ACTIVE_HIGH>;
        vin-supply = <&vcc5v0_device>;
        pinctrl-names = "default";
        pinctrl-0 = <&vcc5v0_host_en>;
};

#使能USB3.1/SATA/PCIe Combo PHY
&combphy1_psu {
        status = "okay";
};

#配置USB3.1 OTG1 Controller
&usb_drd1_dwc3 {
        dr_mode = "host";
        status = "okay";
};
```

4.7 USB PHY 不供电的 DTS 配置

USBDP PHY （OTG0 USB3.1 和 DP 复用）和 ComboPHY1（OTG1 USB3.1 和 PCIe1/SATA1复用）的外部电源可以允许不供电，以降低系统运行功耗和简化硬件电路设计，但需要软件修改对应的 DTS 配置，避免系统启动异常或者 USB 功能异常。

4.7.1 USBDP PHY 不供电的 DTS 配置

```
&usbdp_phy {
        rockchip,usbdpphy-clamp; /* 避免断电影响到总线访问和Logic漏电到USBDP PHY */
        maximum-speed = "high-speed"; /* 限制 u2 only，避免初始化 USBDP PHY */
        status = "okay";
};

&usbdp_phy_dp {
        status = "disabled";
};

&usbdp_phy_u3 {
        status = "disabled";
};
```

内核生效 log

```
rockchip-usbdp-phy 2b010000.phy: Failed to enable usbdpphy because clamp is set
rockchip-usbdp-phy: probe of 2b010000.phy failed with error -95
```

此外，烧写工具必须关闭 USB3 下载功能，否则会出现 Maskrom 下载固件失败的问题，修改方法如下：

Windows 工具的 config.ini 修改：USB3_TRANSFER=FALSE

Linux 工具的 config.ini 修改：usb3_transfer_on=false

4.7.2 ComboPHY1 不供电的 DTS 配置

ComboPHY1 用于 OTG1 的 USB 3.1，但同时也会影响到 OTG1 的 USB2 访问 SoC 总线，因此，需要根据 OTG1 USB2.0 是否使用，进行 DTS 配置。

1. OTG1 USB 3.1 不使用，但 USB 2.0 功能需要使用

```
&combphy1_psu {
        rockchip,dis-u3otg1-port;
        status = "okay";
};

&usb_drd1_dwc3 {
        dr_mode = "host";
        phys = <&u2phy1_otg>, <&combphy1_psu PHY_TYPE_USB3>;  /* 这里必须要引用
combphy1_psu */
        phy-names = "usb2-phy", "usb3-phy";
        maximum-speed = "high-speed";
        snps,dis_u2_susphy_quirk;
        snps,usb2-lpm-disable;
        status = "okay";
};
```

2. OTG1 USB 2.0 和 USB 3.1 均不使用

```
&combphy1_psu {
        status = "disabled";
};

&usb_drd1_dwc3 {
        status = "disabled";
};
```

4.8 Linux USB DT 配置的注意点

4.8.1 USB DT 重要属性说明

4.8.1.1 USB 控制器属性

1. "usb-role-switch" 仅用于标准 Type-C 接口（带有 PD 控制器芯片），同时须配置 dr_mode = "otg" 属性；如果 dr_mode 为非 “otg” 模式，请勿配置 "usb-role-switch" ；
2. "extcon" 属性主要功能之一是动态切换 OTG mode，适用于非标准 Type-C 接口（带有 PD 控制器芯片）的硬件电路设计，如：USB Micro 接口或者 Type-A 接口，同时控制器配置为 “otg” 模式的方案中，实现 OTG 模式动态切换；
3. "snps,dis_u2_susphy_quirk"，关闭 USB 控制器硬件自动 suspend usb2 phy 的功能，主要用于 USB 2.0 only 的方案，以提高 USB 通信的稳定性；
4. "snps,usb2-lpm-disable"，关闭 USB 控制器 Host mode 的 LPM 功能，主要用于 USB 2.0 only 的方案，以提高 USB 外设的兼容性。

4.8.1.2 USB2 PHY 属性

1. "vbus-supply" 和 "phy-supply"，都是用于控制 VBUS 输出 5V，但两者又有使用区别。"vbus-supply" 用于 OTG 口，支持动态开关 VBUS。"phy-supply" 用于 USB HOST 口，系统上电后，VBUS 5V 常开；
2. "rockchip,sel-pipe-phystatus"，该属性用于配置 GRF USB 控制寄存器，选择 GRF 控制 pipe phystatus，替代 USBDP PHY 的控制。主要用于 USB 2.O only 的方案，如果 USBDP PHY 没有使能，必须增加该属性，否则，USB device 无法正常工作；
3. "rockchip,typec-vbus-det"，用于支持 Type-C VBUS_DET 常拉高的硬件设计；
4. "rockchip,dis-u2-susphy"，关闭 USB2 PHY 驱动动态进入 suspend mode 的功能，主要用于 USB 2.0 only 的方案，保持 USB2 PHY 输出时钟给 USB 控制器。

4.8.1.3 USBDP Combo PHY 属性

1. "rockchip,dp-lane-mux" ，非全功能 Type-C 方案中，配置 DP 映射的 Lane number。DP 支持 2 条或 4 条 lane，如 "rockchip,dp-lane-mux = <2, 3>;" 表示 DP Lane0 mapping 至 USBDP PHY 的 Lane2，DP Lane1 mapping 至 USBDP PHY 的 Lane3；同理，"rockchip,dp-lane-mux = <0, 1, 2, 3>;" 表示 DP Lane0 mapping 至 USBDP PHY 的 Lane0 等等，依次类推。

   注意：实际电路中，如果仅支持 USB3.1 但未支持 DP，也需要配置 "rockchip,dp-lane-mux"，否则 USBDP PHY 驱动无法自动分配 lane 给 USB3.1。

2. maximum-speed = "high-speed"; 属性是在USBDP PHY 驱动中将 USB 限制为 USB2.0 only，注意该属性是在父节点 usbdp_phy 中配置。

4.8.1.4 USBC 属性

1. usbc 节点通常作为 i2c 的子节点，除 compatible, reg 等基本属性外，又包括 port 和 connector 两个子节点。其中 port 用于配置与 USB 控制器交互的 role switch 属性；connector 除配置 PD 的基本属性外，还包括 altmodes 和 ports (用于配置 orientation switch 和 altmode mux) 2个字节点。
2. connector 节点中 sink-pdos 及 source-pdos 属性，用于 RK3576 分别作为 Sink 和 Source 时所支持的电压/电流挡位以及其它电源能力的配置。需要注意的是电压/电流挡位要与充电 IC 能力一致。sink-pdos 及 source-pdos 属性字段须遵循 PD Spec，关键字段须用如下文件中的宏定义。

   `include/dt-bindings/usb/pd.h`

5. RK3576 USB OTG mode 切换命令

RK3576 SDK 支持通过软件方法，强制设置 USB OTG 切换到 Host 或者 Peripheral，而不受 USB 硬件电路的 OTG ID 电平或者 Type-C 接口的影响。

RK3576 Linux-6.1 内核切换方式有如下两种：

注意：方式 1 依赖于 USB DTS 的正确配置，只能用于非 Type-C 接口的硬件电路设计，方式 2 没有限制。因此，在不确定软硬件是否正确适配时，推荐使用方式 2。

方式1. [Legacy]

```
#1.Force host mode
  echo host > /sys/devices/platform/2602e000.syscon/2602e000.syscon:usb2-
phy@0/otg_mode
#2.Force peripheral mode
  echo peripheral > /sys/devices/platform/2602e000.syscon/2602e000.syscon:usb2-
phy@0/otg_mode
```

方式2. [New]

```
#1.Force host mode
  echo host > /sys/kernel/debug/usb/23000000.usb/mode
#2.Force peripheral mode
  echo device > /sys/kernel/debug/usb/23000000.usb/mode
```

6. Type-C 控制器芯片支持列表

表 17 Type-C 控制器芯片支持列表

Type-C控制器芯片型号	Linux-6.1	说明
ANX7411	调试中	软件驱动已经支持，功能待进一步调试
AW35615	支持	推荐优先使用，对 DisplayPort Alt Mode 兼容性好 软硬件完全兼容 FUSB302
ET7301B	支持	软硬件完全兼容 FUSB302[note1]
ET7303	支持	硬件兼容 FUSB302，软件驱动与 RT1711 高度相似[note2]
FUSB302	支持	RK平台常用芯片[note2]
HUSB311	支持	RK3576 EVB 默认使用的芯片 硬件兼容 FUSB302，但软件驱动不兼容[note3]
RT1711H	支持	硬件兼容 FUSB302，软件驱动与 ET7303 高度相似[note4]
WUSB3801	不支持	自定义的单线通信机制，误码率高，无法保证通信稳定。

note1.

- Linux-6.1 采用 TCPM 软件框架和 TCPCI 协议，理论上可以兼容所有基于 TCPCI 标准设计的 Type-C 控制器芯片（如：ET7303/HUSB311/RT1711H）；

note2.

- 小封装的 ET7303 (据了解原厂目前没有提供大封装) 已经在 RK 平台验证通过。内核需要单独使能 CONFIG_TYPEC_ET7303。
- DTS 详细配置请参考章节 Type-C USB 3.1/DP 全功能 DTS 配置 中 usbc0 节点，只需要修改该节点名字、reg 地址和 compatible 属性即可。

note3.

- FUSB302 可直接替换为 HUSB311。内核需要单独使用 CONFIG_TYPEC_HUSB311，DTS 配置注意点同上述 note2。

note4.

- RT1711H 硬件兼容 FUSB302/ET7303，同时软件驱动与 ET7303 高度相似。内核需要单独使能 CONFIG_TYPEC_RT1711H，DTS 配置注意点同上述 note2。

7. 参考文档

1. 《Universal Serial Bus Type-C Cable and Connector Specification》
2. 《Universal Serial Bus Power Delivery Specification》
3. 《Rockchip_Developer_Guide_USB_CN》
4. 《Rockchip RK3576 Technical Reference Manual》
5. RK3576 EVB1/EVB2/TEST1/TEST2 硬件原理图