

# Rockchip

# USB 开发指南

发布版本:1.0

日期:2017.02

# 前言

## 概述

## 产品版本

芯片名称	内核版本
RK3328	Linux3.10

## 读者对象

软件工程师，硬件工程师，FAE

## 修订记录

日期	版本	作者	修改说明
2017-02-16	v1.0	WLF WMC MDY	

## 本文档适用范围：

RK3328 芯片，运行 Linux Kernel 3.10 内核。

# 目录

前言 .....	I
目录 .....	II
插图目录 .....	IV
表格目录 .....	V
1 概述 .....	1-1
2 硬件电路及信号 .....	2-1
2.1 USB HOST 控制器硬件电路 .....	2-1
2.1.1 USB 2.0 HOST 控制器硬件电路 .....	2-1
2.1.2 USB 3.0 HOST 控制器硬件电路 .....	2-2
2.2 USB OTG 控制器硬件电路 .....	2-4
2.2.1 USB 2.0 OTG 控制器硬件电路 .....	2-4
3 Kernel 模块配置 .....	3-1
3.1 USB PHY 相关配置 .....	3-1
3.2 USB HOST 相关配置 .....	3-1
3.3 USB OTG 相关配置 .....	3-2
3.4 USB Gadget 配置 .....	3-2
3.5 USB 其它模块配置 .....	3-3
3.5.1 Mass Storage Class (MSC) .....	3-3
3.5.2 USB Serial Converter .....	3-3
3.5.3 USB HID .....	3-4
3.5.4 USB Net .....	3-4
3.5.5 USB Camera .....	3-4
3.5.6 USB Audio .....	3-5
3.5.7 USB HUB .....	3-5
3.5.8 USB 其他设备配置 .....	3-5
4 Device Tree 开发 .....	4-1
4.1 USB PHY DTS .....	4-1
4.1.1 USB 2.0 PHY DTS .....	4-1
4.1.2 USB 3.0 PHY DTS .....	4-2
4.2 USB Controller DTS .....	4-4
4.2.1 USB 2.0 HOST Controller DTS .....	4-4
4.2.2 USB 2.0 OTG Controller DTS .....	4-4
4.2.3 USB 3.0 HOST Controller DTS .....	4-5
5 驱动开发 .....	5-1
5.1 USB PHY drivers .....	5-1
5.1.1 USB 2.0 PHY driver .....	5-1

5.1.2	USB 3.0 PHY driver.....	5-1
5.2	USB Controller drivers .....	5-2
5.2.1	USB 2.0 HOST Controller driver .....	5-2
5.2.2	USB 2.0 OTG Controller driver .....	5-2
5.2.3	USB 3.0 HOST Controller driver .....	5-3
6	Android Gadget 配置.....	6-1
6.1	Gadget 驱动配置.....	6-1
6.2	BOOT IMG 配置.....	6-1
7	常见问题分析.....	7-3
7.1	设备枚举日志 .....	7-3
7.1.1	USB 2.0 OTG 正常开机日志 .....	7-3
7.1.2	USB 2.0 OTG Device 正常连接日志 .....	7-3
7.1.3	USB 2.0 OTG Device 正常断开日志 .....	7-3
7.1.4	USB 2.0 OTG HOST 设备正常连接日志 .....	7-4
7.1.5	USB 2.0 HOST 设备正常连接日志 .....	7-4
7.1.6	USB 2.0 HOST-LS/FS/HS 设备断开日志 .....	7-5
7.1.7	USB 3.0 HOST-SS 设备正常连接日志 .....	7-5
7.2	USB 常见问题分析.....	7-6
7.2.1	软件配置.....	7-6
7.2.2	硬件电路.....	7-6
7.2.3	Device 功能异常分析.....	7-6
7.2.4	Host 功能异常分析.....	7-7
7.2.5	USB Camera 异常分析.....	7-8
7.2.6	USB 充电检测 .....	7-8
7.3	PC 驱动问题 .....	7-10
8	USB 信号测试.....	8-1

# 插图目录

图表 1-1 USB OTG 2.0 Architecture .....	1-1
图表 1-2 USB Host 2.0 Controller Block Diagram .....	1-2
图表 1-3 USB2.0 PHY Block Diagram .....	1-2
图表 1-4 USB Host 3.0 Block Diagram .....	1-3
图表 1-5 USB PHY 3.0 Block Diagram .....	1-4
图表 2-1 USB 2.0 HOST SoC 信号引脚及供电电源 .....	2-1
图表 2-2 USB 2.0 HOST VBUS 5V 控制电路.....	2-1
图表 2-3 USB Host 2.0 Standard-A 接口电路.....	2-2
图表 2-4 USB 3.0 Standard-A plug .....	2-2
图表 2-5 USB 3.0 Standard-A receptacle .....	2-2
图表 2-6 USB3.0 Standard-A 引脚分配 .....	2-3
图表 2-7 USB3.0 控制器 SoC 信号引脚以及供电电源电路 .....	2-3
图表 2-8 USB3.0 Standard-A 接口电路 .....	2-4
图表 2-9 USB3.0 VBUS 5V 控制电路 .....	2-4
图表 2-10 USB OTG 2.0 控制器 SoC 信号引脚以及供电电源电路 .....	2-5
图表 2-11 USB OTG 2.0 Standard-A 接口电路 .....	2-5
图表 2-12 USB OTG 2.0 VBUS 5V 控制电路 .....	2-6
图表 3-1 USB PHY Driver 配置示意图 .....	3-1
图表 3-2 USB 2.0 OTG 内核配置选项 .....	3-2
图表 3-3 USB Gadget Driver 内核配置 .....	3-2
图表 3-4 SCSI 配置 .....	3-3
图表 3-5 MSC 配置 .....	3-3
图表 4-1 图 USB 2.0 PHY DTSI 配置示意图 .....	4-1
图表 4-2 USB 2.0 PHY VBUS Regulator 配置示意图 .....	4-2
图表 4-3 USB 2.0 PHY VBUS Pinctrl 配置示意图 .....	4-2
图表 4-4 USB 2.0 PHY DTS 配置示意图 .....	4-2
图表 4-5 USB 3.0 PHY DTSI 配置 .....	4-3
图表 4-6 EHCI DTSI 配置示意图 .....	4-4
图表 4-7 OHCI DTSI 配置示意图 .....	4-4
图表 4-8 USB 2.0 OTG 控制器节点的 DTSI 配置 .....	4-5
图表 4-9 USB 2.0 OTG PHY 节点的 DTSI 配置 .....	4-5
图表 4-10 USB 3.0 Host DTSI 配置 .....	4-6
图表 6-1 Android Gadget init.rk30board.usb.rc 的 usb 配置信息 .....	6-2
图表 7-1 USB 充电检测流程 .....	7-9
图表 7-2 SDP 检测波形 .....	7-9
图表 7-3 DCP 检测波形 .....	7-10

# 表格目录

表 1-1 RK3399 平台 USB 控制器列表 .....	1-1
---------------------------------	-----

# 1 概述

Rockchip SOC 通常内置多个 USB 控制器，不同控制器互相独立，请在芯片 TRM 中获取详细信息。由于部分 USB 控制器有使用限制，所以请务必明确方案的需求及控制器限制后，再确定 USB 的使用方案。RK3328 芯片内置的 USB 控制器如表 1-1 所示：

表 1-1 RK3328 平台 USB 控制器列表

控制器 芯片	USB2.0 OTG (EHCI&OHCI)	USB2.0 HOST (EHCI&OHCI)	USB3.0 HOST (XHCI)
RK3328	√	√	√

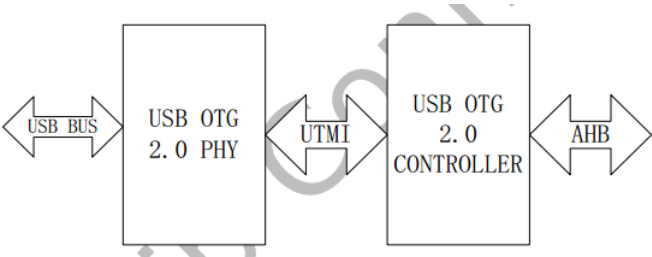
RK3328 SoC 包含 1 个 USB2.0 OTG 控制器，1 个 USB2.0 HOST 控制器，1 个 USB3.0 HOST 控制器，各个控制器及 USB PHY 的具体硬件信息说明如下：

■ USB OTG 2.0

USB OTG 2.0 is a Dual-Role Device controller, which supports both device and host functions and is fully compliant with OTG Supplement to USB2.0 specification and support high-speed(480Mbps),full-speed(12Mbps),low-speed(1.5Mbps) transfer.

Features:

- 1. pliant with the OTG Supplement to the USB2.0 Specification
- 2. rates in High-Speed and Full-Speed mode
- 3. ort 9 channels in host mode
- 4. Device mode endpoints in addition to control endpoint 0, 4 in, 3 out and 2 IN/OUT
- 5. It-in one 1024x35 bits FIFO
- 6. kernal DMA with scatter/gather function
- 7. ports packet-based, dynamic FIFO memory allocation for endpoints for flexible,
- 8. licient use of RAM
- 9. port dynamic FIFO sizing



图表 1-1 USB OTG 2.0 Architecture

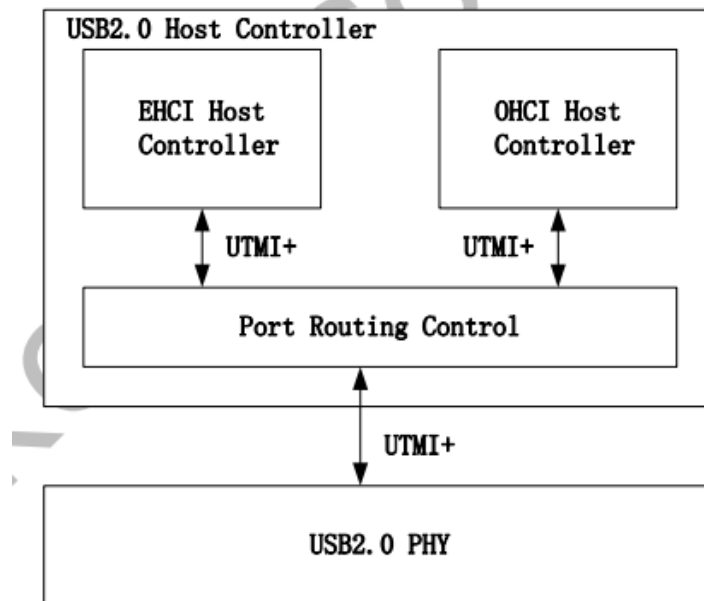
■ USB Host 2.0

Embedded 1 USB Host 2.0 interfaces

Features:

- 1. Compatible Specification
- 2. Universal Serial Bus Specification, Revision 2.0
- 3. Enhanced Host Controller Interface Specification(EHCI), Revision 1.0

4. Open Host Controller Interface Specification(OHCI), Revision 1.0a
5. Support high-speed(480Mbps), full-speed(12Mbps) and low-speed(1.5Mbps)

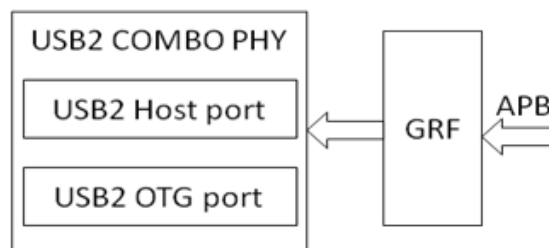


图表 1-2 USB Host 2.0 Controller Block Diagram

#### ■ USB PHY 2.0

Host port: used for USB2.0 Host controller

OTG Port: used for USB2.0 OTG controller



图表 1-3 USB2.0 PHY Block Diagram

#### ■ USB Host 3.0

USB 3.0 Host Controller can act as static host USB 2.0/3.0. It can perform data transmission between host and device as host for Super-Speed / High-Speed / Full-Speed / Low-Speed

Features:

##### ● General Features

##### 1. Compatible Specification

Universal Serial Bus 3.0 Specification, Revision 1.0

Universal Serial Bus Specification, Revision 2.0

eXtensible Host Controller Interface for Universal Serial Bus (xHCI), Revision 1.1

##### 2. Support Control/Bulk(including stream)/Interrupt/Isochronous Transfer

##### 3. Simultaneous IN and OUT transfer for USB3.0, up to 8Gbps bandwidth

##### 4. Descriptor caching and data pre-fetching used to improve system performance

in

high-latency systems



5. LPM protocol in USB 2.0 and U0, U1, U2, and U3 states for USB 3.0
6. Dynamic FIFO memory allocation for endpoints
7. Keep-Alive feature in LS mode and (micro-)SOFs in HS/FS modes
8. Low MIPS requirement

Driver involved only in setting up transfers and high-level error recovery

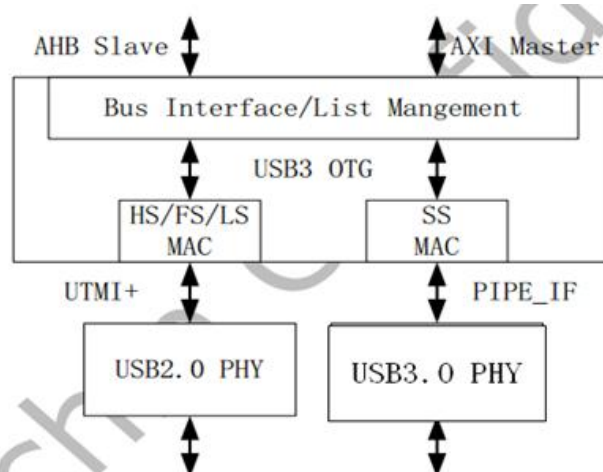
Hardware handles data packing and routing to a specific pipe

- USB Class-Specific Device Features

1. Stream support for UASP application
2. Gathering of scattered packet to support Ethernet Over USB
3. Scheduling of multiple Ethernet packets without interrupt
4. Variable FIFO buffer allocation for each endpoint
5. For isochronous applications, scheduling of variable-length payloads for each microframe
6. Microframe precise scheduling for isochronous applications
7. Configurable endpoint type selection and dynamic FIFO allocation to facilitate multi-function/composite device implementation. During set-config or alternate-setting, device resources are reconfigured to meet the configuration or alternate setting requirements.

- USB 3.0 xHCI Host Features

1. Support up to 64 devices
2. Support 1 interrupter
3. Support 1 USB2.0 port and 1 Super-Speed port
4. Support xHCI Debug Capability
5. Concurrent USB3.0/USB2.0 traffic, up to 8.48Gbps bandwidth
6. Support standard or open-source xHCI and class drive



图表 1-4 USB Host 3.0 Block Diagram

- USB PHY 3.0

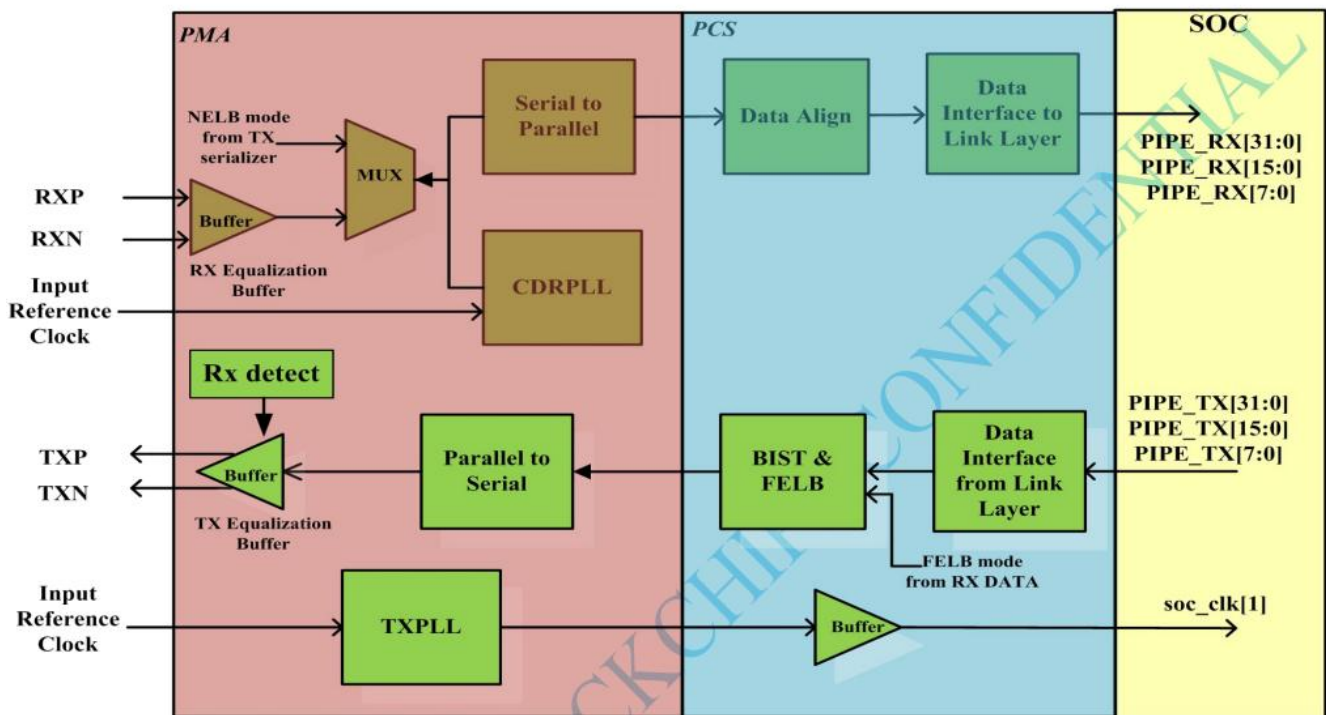
The USB SuperSpeed PHY Layer handles the low level USB SuperSpeed protocol and signaling. This includes features such as: Data serialization and de-serialization, 8b/10b encoding, analog buffers, elastic buffers and receiver detection.

Features:

1. Supports 5.0Gb/s serial data transmission rate
2. Utilizes 8-bit, 16-bit or 32-bit parallel interface to transmit and receive USB

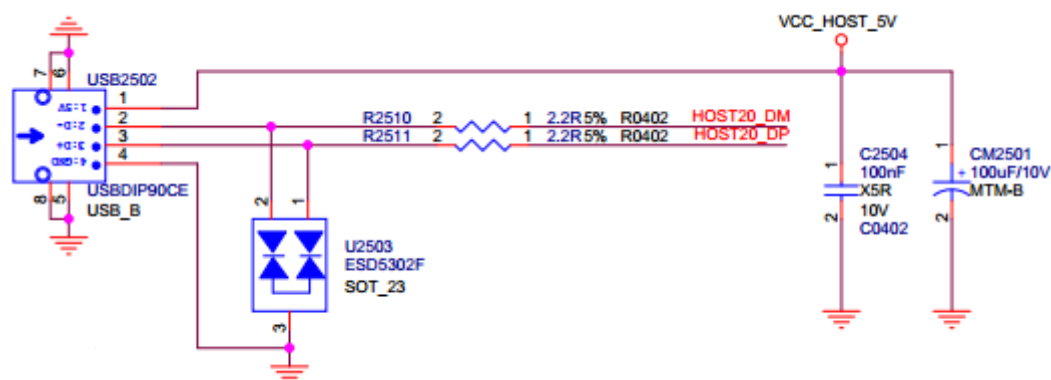
SuperSpeed data

3. Allows integration of high speed components into a single functional block as seen by the device designer.
4. Data and clock recovery from serial stream on the USB SuperSpeed bus
5. Holding registers to stage transmit and receive data
6. Supports direct disparity control for use in transmitting compliance pattern
7. 8b/10b encode/decode and error indication
8. Receiver detection
9. Low Frequency Periodic Signaling (LFPS) Transmission



图表 1-5 USB PHY 3.0 Block Diagram

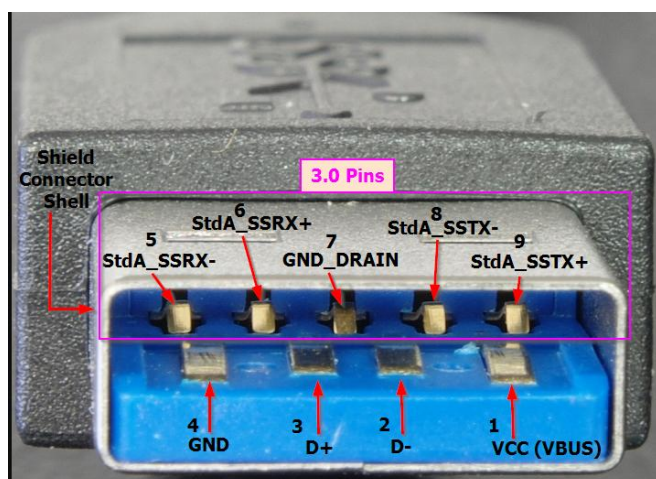




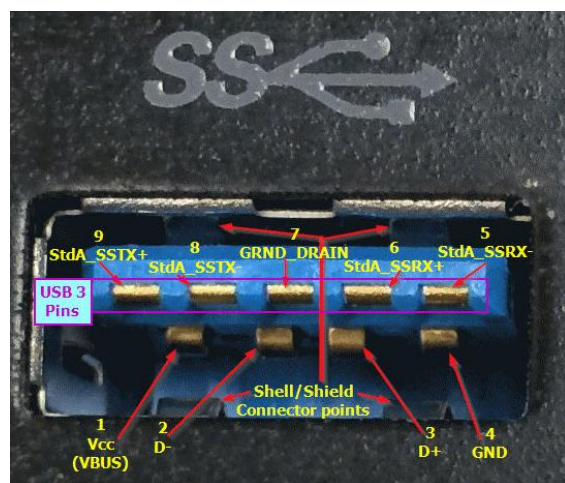
图表 2-3 USB Host 2.0 Standard-A 接口电路

### 2.1.2 USB 3.0 HOST 控制器硬件电路

RK3328 支持 USB3.0 Host 功能，且向下兼容 USB2.0 Host 功能，最大传输速率为 5Gbps，物理接口为 USB3.0 Type-A，如图 2-4 和图 2-5 所示，在传输线方面，USB3.0 支持长达 3 米的四线差分信号线及 11 英寸 PCB。5Gbps 信号在长线缆上采用的是差分信号方式传输，从而避免信号被干扰及减少电磁干扰问题。



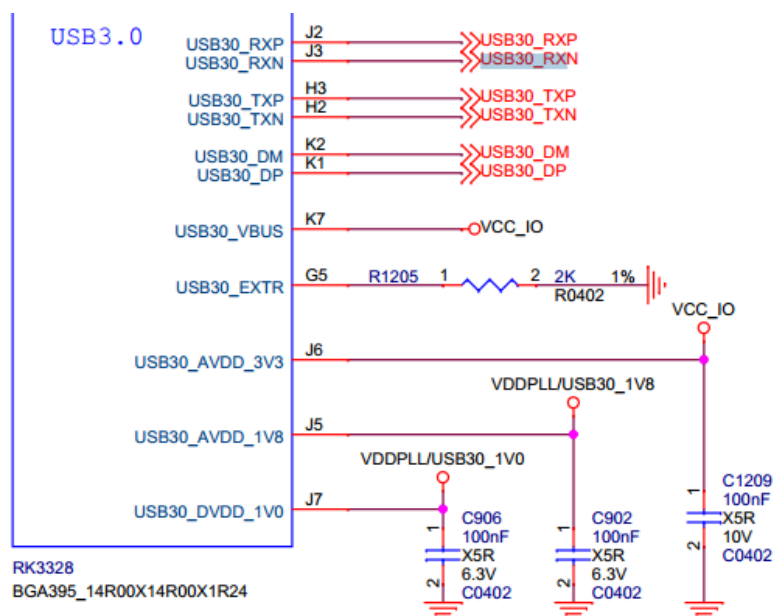
图表 2-4 USB 3.0 Standard-A plug



图表 2-5 USB 3.0 Standard-A receptacle

Pin	Color	Signal name		Description
		A connector	B connector	
Shell	N/A	Shield		Metal housing
1	Red	VBUS		Power
2	White	D-		USB 2.0 differential pair
3	Green	D+		
4	Black	GND		Ground for power return
5	Blue	StdA_SSRX-	StdB_SSTX-	SuperSpeed transmitter differential pair
6	Yellow	StdA_SSRX+	StdB_SSTX+	
7	N/A	GND_DRAIN		Ground for signal return
8	Purple	StdA_SSTX-	StdB_SSRX-	SuperSpeed receiver differential pair
9	Orange	StdA_SSTX+	StdB_SSRX+	

图表 2-6 USB3.0 Standard-A 引脚分配

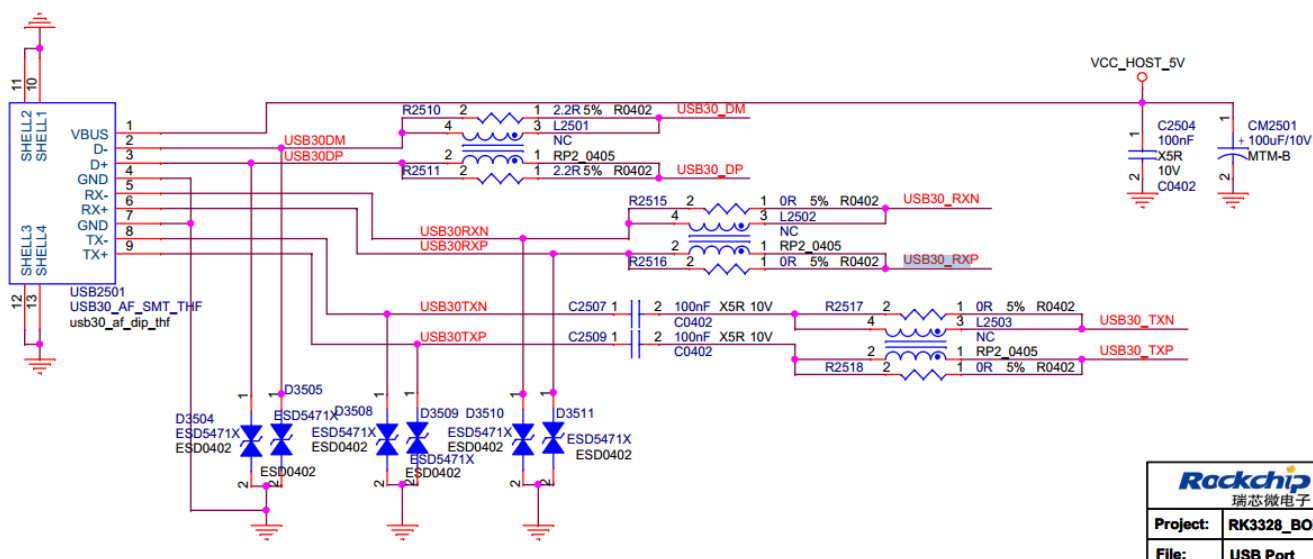


图表 2-7 USB3.0 控制器 SoC 信号引脚以及供电电源电路

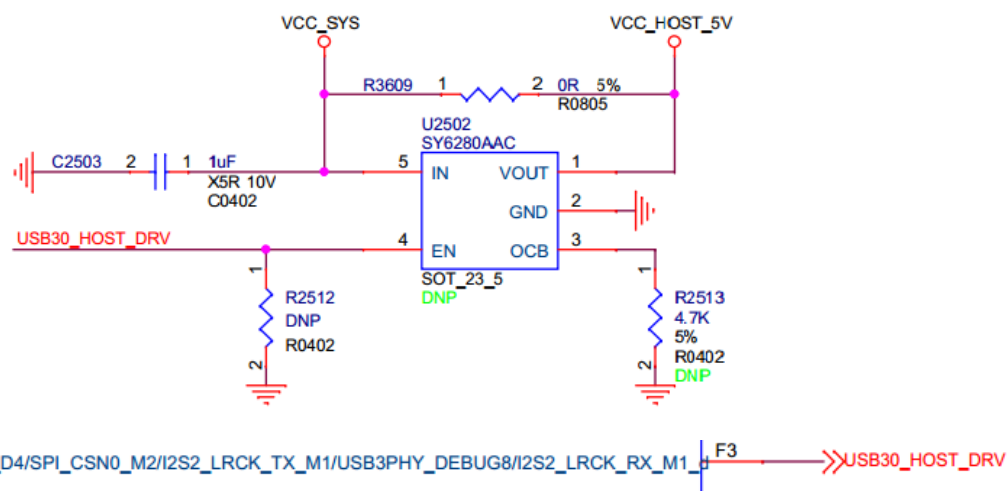
Note:

USB30\_AVDD\_3V3, USB30\_AVDD\_1V8, USB30\_DVDD\_1V0 为 USB 3.0 HOST 控制器及 PHY 的供电电源, 必须保证电源电压纹波低于 10%;

USB30\_EXTR 为 USB 3.0 PHY 的外部基准电阻, 必须为 2k  $\Omega$ ;



图表 2-8 USB3.0 Standard-A 接口电路



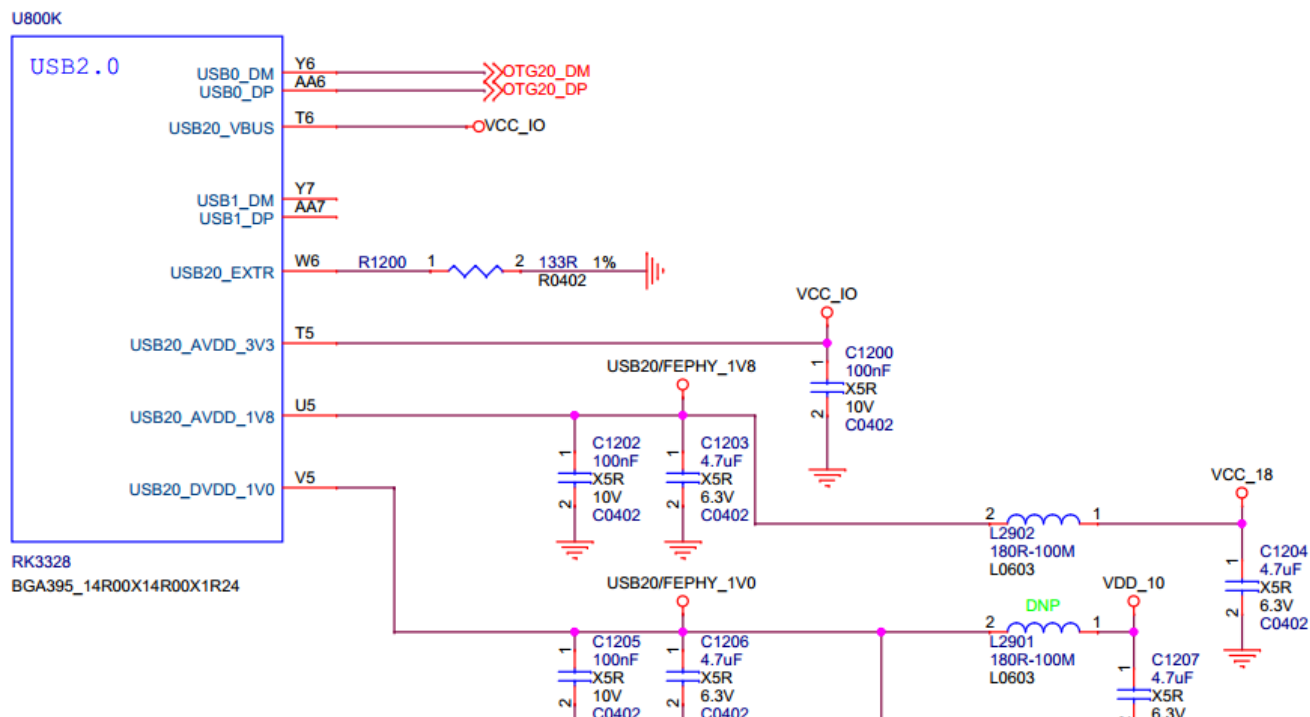
图表 2-9 USB3.0 VBUS 5V 控制电路

## 2.2 USB OTG 控制器硬件电路

### 2.2.1 USB 2.0 OTG 控制器硬件电路

RK3328 SoC 含有一个 USB2.0 OTG 控制器，可支持 Peripheral 及 Host 模式。接口采用 TYPE-A 接口，硬件电路如图 2-10 所示。

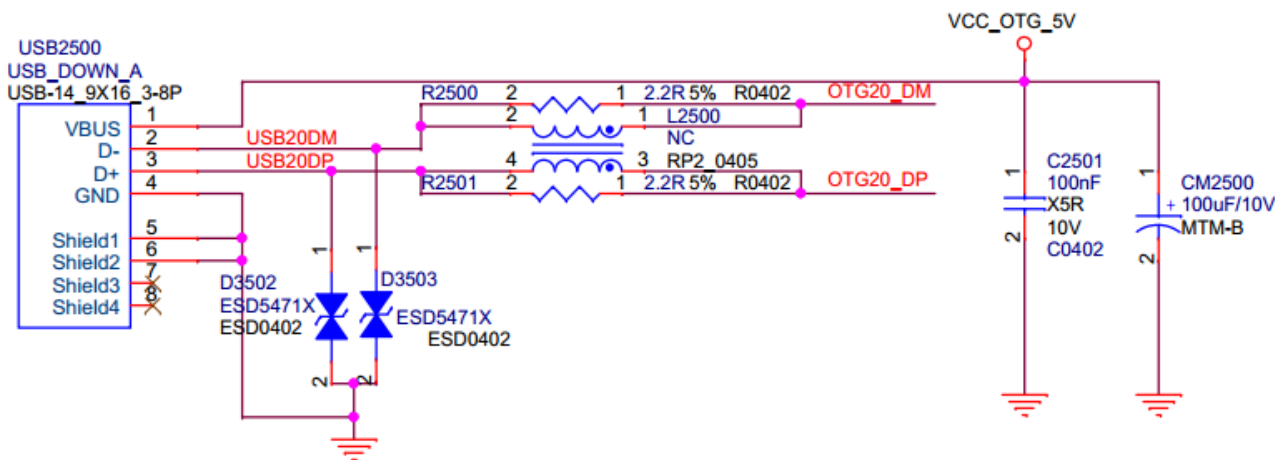




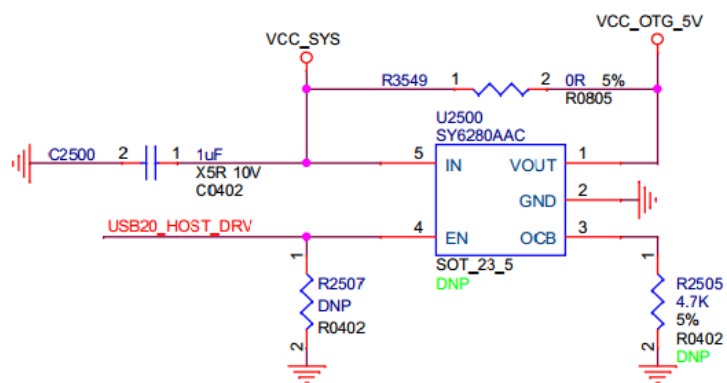
图表 2-10 USB OTG 2.0 控制器 SoC 信号引脚以及供电电源电路

Note:

1. USB20\_AVDD\_3V3, USB20\_AVDD\_1V8, USB20\_DVDD\_1V0 为 USB OTG 2.0 和 USB Host 2.0 控制器及 PHY 的供电电源, 必须保证电源电压纹波低于 10%;
2. USB20\_EXTR 为 USB PHY 2.0 的外部基准电阻, 必须为 133R;



图表 2-11 USB OTG 2.0 Standard-A 接口电路



GPIO3\_A2/TSP\_CLK/CIF\_CLKIN/SDMMC0EXT\_CLK/SPI\_RXD\_M2/USB3PHY\_DEBUG3/I2S2\_SDI\_M1 E1 USB20\_HOST\_DRV

图表 2-12 USB OTG 2.0 VBUS 5V 控制电路



# 3 Kernel 模块配置

USB 模块的配置及保存和其它内核模块的配置方法一样：

- 导入默认配置

```
make ARCH=arm64 rockchip_linux_defconfig
```

- 选择 kernel 配置

```
make ARCH=arm64 menuconfig
```

- 保存 default 配置

```
make ARCH=arm64 savedefconfig
```

保存 default 配置，然后用 defconfig 替换 rockchip\_linux\_defconfig。

## 3.1 USB PHY 相关配置

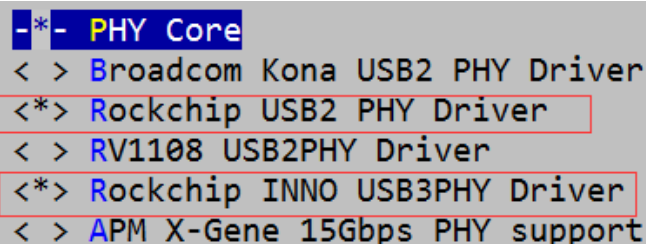
USB PHY 模块的配置位于

```
Device Drivers --->
```

```
PHY Subsystem --->
```

RK3328 USB2.0 PHY 使用的是 Innosilicon USB2.0 PHY IP，所以应选择“Rockchip USB2 PHY Driver”。

RK3328 USB3.0 PHY 使用的是 Innosilicon USB3.0 PHY IP，所以应选择“Rockchip INNO USB3PHY Driver”。



```
-*- PHY Core
< > Broadcom Kona USB2 PHY Driver
<*> Rockchip USB2 PHY Driver
< > RV1108 USB2PHY Driver
<*> Rockchip INNO USB3PHY Driver
< > APM X-Gene 15Gbps PHY support
```

图表 3-1 USB PHY Driver 配置示意图

## 3.2 USB HOST 相关配置

USB 模块的配置位于

```
Device Drivers --->
```

```
[*] USB support --->
```

必须选上 USB Support 项后才能支持 USB 模块并进行进一步的配置。

需要支持 USB HOST，首先需要选上<\*>Support for Host-side USB 项，然后会出现如下的 HOST 相关的配置，其中，HOST1.1 选择 OHCI Driver 配置，HOST2.0 选择 EHCI Driver 配置，HOST3.0 选择 XHCI Driver 配置。

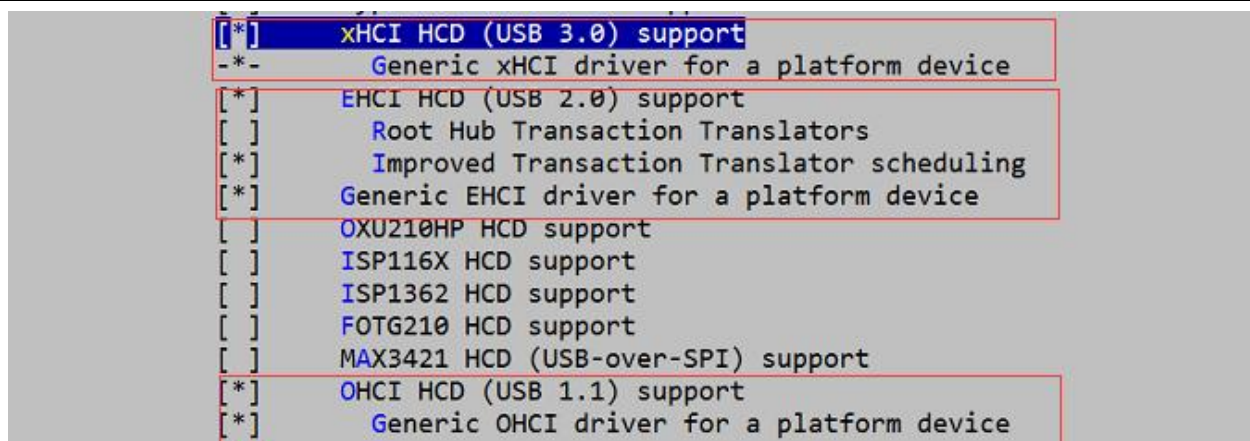


图 3-1 Host Driver 内核的配置选项

### 3.3 USB OTG 相关配置

Device Drivers --->

[\*] USB support --->

ROCKCHIP USB Support --->

```
< > Rockchip USB 2.0 host controller
<*> RockChip USB 2.0 OTG controller
```

图表 3-2 USB 2.0 OTG 内核配置选项

### 3.4 USB Gadget 配置

Device Drivers --->

[\*] USB support --->

<\*> USB Gadget Support --->

```
-- USB Gadget Support
[ ] Debugging messages (DEVELOPMENT)
[*] Debugging information files (DEVELOPMENT)
[ ] Debugging information files in debugfs (DEVELOPMENT)
(500) Maximum VBUS Power usage (2-500 mA)
(2) Number of storage pipeline buffers
USB Peripheral Controller --->
<*> USB Gadget Drivers (Android Composite Gadget) --->
    Android Composite Gadget
[ ] Use double word aligned
[*] USB Mass storage csw hack Feature
[ ] USB MSC performance profiling
```

图表 3-3 USB Gadget Driver 内核配置

Note: RK3328 目前支持 MTP、PTP、Accessory、ADB、Audio、ACM 等 Gadget 功能。

## 3.5 USB 其它模块配置

### 3.5.1 Mass Storage Class (MSC)

U 盘属于 SCSI 设备，所以在配置 USB 模块之前需要配置 SCSI 选项（默认配置已经选上）。

```
Device Drivers --->
  SCSI device support --->
    <*> SCSI disk support

[*] SCSI device support
[ ] SCSI: use blk-mq I/O path by default
[*] legacy /proc/scsi/ support
    *** SCSI support type (disk, tape, CD-ROM) ***
[*] SCSI disk support → 支持U 盘
[ ] SCSI tape support
[ ] SCSI OnStream SC-x0 tape support
[ ] SCSI CDROM support → 支持USB CD-ROM
[*] SCSI generic support
[*] SCSI media changer support
[*] Verbose SCSI error reporting (kernel size +=75K)
[*] SCSI logging facility
[*] Asynchronous SCSI scanning
```

图表 3-4 SCSI 配置

配置完 SCSI Device Support 后，可以在 USB Support 中找到如下选项，选上即可。

```
[*] USB Mass Storage support
[ ] USB Mass Storage verbose debug
```

图表 3-5 MSC 配置

### 3.5.2 USB Serial Converter

- 支持 USB 3G Modem

USB 3G Modem 使用的是 USB 转串口，使用时需要选上如下选项：

```
[*] USB Serial Converter support --->
  [*] USB driver for GSM and CDMA modems
```

- 支持 PL2303

如果要使用 PL2303，输出数据到串口，需要选择如下选项：

```
[*] USB Serial Converter support --->
  [*] USB Prolific 2303 Single Port Serial Driver
```

- 支持 USB GPS

如果要支持 USB GPS，如 u-blox 6 - GPS Receiver 设备，需要选择如下选项：

```
Device Drivers --->
  [*] USB support --->
    [*] USB Modem (CDC ACM) support
```

### 3.5.3 USB HID

USB 键鼠的配置选项如下:

```
Device Drivers --->
  HID support --->
    USB HID support --->
      [*] USB HID transport layer
      [ ] PID device support
      [*] /dev/hiddev raw HID device support
```

### 3.5.4 USB Net

- USB Bluetooth

```
[*] Networking support --->
  [*] Bluetooth subsystem support --->
    Bluetooth device drivers --->
      [*] HCI USB driver
```

- USB Wifi

通常直接使用 Vendor 提供的驱动和配置。

- USB Ethernet

```
Device Drivers --->
  [*] Network device support --->
    [*] USB Network Adapters --->
      [*] USB CATC NetMate-based Ethernet device support
      [*] USB KLSI KL5USB101-based ethernet device support
      [*] USB Pegasus/Pegasus-II based ethernet device support
      [*] USB RTL8150 based ethernet device support
      [*] Realtek RTL8152/RTL8153 Based USB Ethernet Adapters
      [ ] Microchip LAN78XX Based USB Ethernet Adapters
      [*] Multi-purpose USB Networking Framework
      [*] ASIX AX88xxx Based USB 2.0 Ethernet Adapters
      [*] ASIX AX88179/178A USB 3.0/2.0 to Gigabit Ethernet
      *- CDC Ethernet support (smart devices such as cable modems)
      [*] CDC EEM support
      *- CDC NCM support
```

### 3.5.5 USB Camera

```
Device Drivers --->
  [*] Multimedia support --->
    [*] Media USB Adapters --->
```

```
--- Media USB Adapters
    *** Webcam devices ***
[*] USB Video Class (UVC)
[*] UVC input events device support
```

### 3.5.6 USB Audio

```
Device Drivers --->
[*] Sound card support --->
    [*] Advanced Linux Sound Architecture --->
        [*] USB sound devices --->
            [*] USB Audio/MIDI driver
```

### 3.5.7 USB HUB

如果要支持 USB HUB，请将“Disable external HUBs”配置选项去掉。

```
Device Drivers --->
[*] USB support --->
    [ ] Disable external hubs
```

### 3.5.8 USB 其他设备配置

其他有可能用到的 USB 设备还有很多，如 GPS，Printer 等，有可能需要 Vendor 定制的驱动，也有可能是标准的 Class 驱动，如需支持，可直接在网络上搜索 Linux 对该设备支持要做的工作，RK 平台并无特殊要求，可直接参考。



# 4 Device Tree 开发

ARM Linux 内核在 Linux-3.x 内核取消了传统的设备文件而用设备树（DT）取代，因此，现在在内核有关硬件描述的信息都需要放入 DT 中配置，下面对涉及到 USB 模块的 DT 开发做以详细说明。

## 4.1 USB PHY DTS

USB2.0 PHY 的配置主要包括 PHY 的时钟、中断配置和 Vbus Supply 的配置。

USB3.0 PHY 的配置主要包括 PHY 的时钟、中断配置、Reset 和 Vbus Supply 的配置。

### 4.1.1 USB 2.0 PHY DTS

USB2.0 PHY 详细配置可参考内核文档：

Documentation/devicetree/bindings/phy/phy-rockchip-usb2.txt

具体分为 DTSI 和 DTS 两部分配置，下面已 RK3328 的一个 Host Port 的 PHY 为例说明。

如下图 4-1 所示，为 DTSI 的配置，DTSI 主要配置 PHY 的公有属性。

```
usb2phy_grf: syscon@fff450000 {
    compatible = "rockchip,rk322xh-usb2phy-grf",
                 "rockchip,usb2phy-grf", "syscon", "simple-mfd";
    reg = <0x0 fff450000 0x0 0x1000>;

    u2phy: usb2-phy@104 {
        compatible = "rockchip,rk322xh-usb-phy";
        #address-cells = <1>;
        #size-cells = <0>;
        status = "disabled";

        u2phy_host: host-port {
            #phy-cells = <0>;
            reg = <0x104>;
            interrupts = <GIC_SPI 62 IRQ_TYPE_LEVEL_HIGH>;
            interrupt-names = "linestate";
        };
    };
};
```

图表 4-1 图 USB 2.0 PHY DTSI 配置示意图

首先，USB PHY Driver 中都是在操作 GRF，所以 USB PHY 的节点必须作为 GRF 的一个子节点。

其次，USB PHY 节点中包括 USB PHY 的硬件属性和 PHY port 的硬件属性，其中 PHY 的属性为所有 port 的共有属性，比如 Input 时钟；Port 属性主要包括各个 port 所拥有的中断，比如 Linestate 中断。

最后，需要注意的是 PORT 的名称，HOST 对应的 port 要求命名为“host-port”，OTG 对应的命名为“otg-port”，因为 Driver 中根据这两个名称做不同 port 的初始化。

DTS 的配置，主要根据不同的产品形态，配置 PHY 的私有属性。目前 SDK-DTS 的配置，主要包括 phy-port 的 Enable 以及 phy-Supply 即 Vbus Supply 的配置。下面给出一个配置 Vbus Supply 的参考（有些产品形态 Vbus 5V 为常供电，不需要 DTS 的配置）。

Vbus supply 的配置一般有两种方式，一种是配置成 GPIO 形式，直接在驱动中控制

GPIO，进而控制的供给；另外一种是目前内核比较通用的 Regulator 配置，当前的 USB 2.0 PHY 只支持第二种方法，即使用 Regulator 配置。以 Host Vbus 的配置，说明 regulator 的配置方法。其主要分为 Regulator 及 pinctrl 两个节点的配置。

```
vbus_host: vbus-host-regulator {
    compatible = "regulator-fixed";
    enable-active-high;
    gpio = <&gpio4 25 GPIO_ACTIVE_HIGH>;
    pinctrl-names = "default";
    pinctrl-0 = <&host_vbus_drv>;
    regulator-name = "vbus_host";
};
```

图表 4-2 USB 2.0 PHY VBUS Regulator 配置示意图

```
usb2 {
    host_vbus_drv: host-vbus-drv {
        rockchip,pins =
            <4 25 RK_FUNC_GPIO &pcfg_pull_none>;
    };
};
```

图表 4-3 USB 2.0 PHY VBUS Pinctrl 配置示意图

如上图 4-2 所示，这是一个 vbus-host-regulator 的配置实例，“enable-active-high”属性标识 GPIO 拉高使能；“pinctrl-0 = <&host\_vbus\_drv>”Property 代表这个 regulator 所引用的 Pinctrl 中节点的名称，具体 Regulator 的配置可参考 Linux Kernel 相关 Regulator 的文档。通过对于 USB 模块而言，vbus-regulator 应该在 DTS 中（而不是 DTSI 中）做配置。如上图 4-3 所示，这是 host vbus-drv 的 pinctrl 属性，rockchip,pins 属性即 GPIO 信息，需要从硬件原理图获知。这个节点作为 Pinctrl 的子节点，通过在 DTSI（而不是 DTS 中）做配置。

在配置完 Regulator 及 pinctrl 两个节点后，USB2 PHY port 就可以引用节点，对 Vbus 的属性“phy-supply”进行配置。

```
&u2phy_host {
    phy-supply = <&vbus_host>;
    status = "okay";
};
```

图表 4-4 USB 2.0 PHY DTS 配置示意图

### 4.1.2 USB 3.0 PHY DTS

RK3328 USB3.0 PHY 为 Innosilicon USB3.0 IP，详细的配置说明请查看：  
Documentation/devicetree/bindings/phy/phy-rockchip-inno-usb3.txt  
Example:

```

u3phy: usb3-phy@ff470000 {
    compatible = "rockchip,rk322xh-u3phy";
    reg = <0x0 0xff470000 0x0 0x0>;
    rockchip,u3phygrf = <&usb3phy_grf>;
    rockchip,grf = <&grf>;
    clocks = <&clk_gates28 1>, <&clk_gates28 2>;
    clock-names = "usb3phy-otg", "usb3phy-pipe";
    interrupts = <GIC_SPI 77 IRQ_TYPE_LEVEL_HIGH>;
    interrupt-names = "linestate";
    resets = <&reset RK322XH_SRST_USB3PHY_U2>,
            <&reset RK322XH_SRST_USB3PHY_U3>,
            <&reset RK322XH_SRST_USB3PHY_PIPE>,
            <&reset RK322XH_SRST_USB3OTG_UTMI>,
            <&reset RK322XH_SRST_USB3PHY_OTG_P>,
            <&reset RK322XH_SRST_USB3PHY_PIPE_P>;
    reset-names = "u3phy-u2-por", "u3phy-u3-por",
                  "u3phy-pipe-mac", "u3phy-utmi-mac",
                  "u3phy-utmi-apb", "u3phy-pipe-apb";
    usb30-drv-gpio = <&gpio0 GPIO_A0 GPIO_ACTIVE_LOW>;
    #address-cells = <2>;
    #size-cells = <2>;
    ranges;
    status = "disabled";
}

```

```

u3phy_utmi: utmi@ff470000 {
    reg = <0x0 0xff470000 0x0 0x8000>;
    #phy-cells = <0>;
    status = "disabled";
};

u3phy_pipe: pipe@ff478000 {
    reg = <0x0 0xff478000 0x0 0x8000>;
    #phy-cells = <0>;
    status = "disabled";
};
};

```

图表 4-5 USB 3.0 PHY DTSI 配置

DTSI 重要属性说明:

reset-names :

- \* "u3phy-u2-por" for the USB 2.0 logic of USB 3.0 PHY
- \* "u3phy-u3-por" for the USB 3.0 logic of USB 3.0 PHY
- \* "u3phy-pipe-mac" for the USB 3.0 PHY pipe MAC
- \* "u3phy-utmi-mac" for the USB 3.0 PHY utmi MAC
- \* "u3phy-utmi-apb" for the USB 3.0 PHY utmi apb
- \* "u3phy-pipe-apb" for the USB 3.0 PHY pipe apb
- \* "u3phy\_utmi" : USB 2.0 utmi phy.
- \* "u3phy\_pipe" : USB 3.0 pipe phy.

其中, "usb30-drv-gpio"用于控制 USB 3.0 的 VBUS 5V 输出, 需要根据实际的硬件设计进行配置。其余属性建议不要修改。



## 4.2 USB Controller DTS

USB2.0 控制器主要包括 EHCI、OHCI、OTG。其中 EHCI 和 OHCI Rockchip 采用 Linux 内核 Generic 驱动，一般开发时只需要对 DT 作相应配置，即可正常工作。

### 4.2.1 USB 2.0 HOST Controller DTS

如下图 4-6 所示，为 RK3328 上一个 EHCI 控制器的典型配置，主要包括 register、interrupts、clocks 的配置。需要注意，EHCI 相关的时钟，通常需要配置 EHCI 控制器和 EHCI/OHCI 仲裁器两个时钟。此外，phys 直接配置对应 phy-port 的名称即可。

```
usb_ehci: usb@ff5c0000 {
    compatible = "generic-ehci";
    reg = <0x0 0xff5c0000 0x0 0x10000>;
    interrupts = <GIC_SPI 16 IRQ_TYPE_LEVEL_HIGH>;
    clocks = <&clk_gates19 6>, <&clk_gates19 7>;
    clock-names = "hclk_host0", "hclk_host0_arb";
    phys = <&u2phy_host>;
    phy-names = "usb";
    status = "disabled";
};
```

图表 4-6 EHCI DTSI 配置示意图

如下图 4-7 所示，为 RK3328 上一个 OHCI 控制器的配置，其内容基本跟 EHCI 相同。

```
usb_ohci: usb@ff5d0000 {
    compatible = "generic-ohci";
    reg = <0x0 0xff5d0000 0x0 0x10000>;
    interrupts = <GIC_SPI 17 IRQ_TYPE_LEVEL_HIGH>;
    phys = <&u2phy_host>;
    phy-names = "usb";
    status = "disabled";
};
```

图表 4-7 OHCI DTSI 配置示意图

### 4.2.2 USB 2.0 OTG Controller DTS

USB 2.0 OTG DTS 包含 “usb2\_otg” 和 “dwc\_control\_usb” 两个节点。其中，“usb2\_otg” 对应 OTG 控制器的硬件信息，而 “dwc\_control\_usb” 对应 USB 2.0 PHY 的硬件信息，“dwc\_control\_usb” 节点中包括一个充电检测子节点 “usb\_bc”。节点中涉及相关硬件信号，如中断、时钟等可参阅 TRM 手册，图 4-8 和图 4-9 为 SDK 参考配置。

详细的配置说明，请参考文档：

Documentation/devicetree/bindings/usb/rockchip-usb.txt

```
usb2_otg: usb@ff580000 {
    compatible = "rockchip,rk322xh_usb20_otg";
    reg = <0x0 0xff580000 0x0 0x40000>;
    interrupts = <GIC_SPI 23 IRQ_TYPE_LEVEL_HIGH>;
    clocks = <&clk_gates17 14>, <&clk_gates19 8>, <&clk_gates19 9>;
    clock-names = "pclk_usb2grf", "hclk_otg", "hclk_otg_pmu";
    resets = <&reset RK322XH_SRST_USB20TG_H>,
            <&reset RK322XH_SRST_USB20TG_UTMI>,
            <&reset RK322XH_SRST_USB20TG>;
    reset-names = "otg_ahb", "otg_phy", "otg_controller";
    /*0 - Normal, 1 - Force Host, 2 - Force Device*/
    rockchip,usb-mode = <0>;
    status = "disabled";
};
```

图表 4-8 USB 2.0 OTG 控制器节点的 DTSI 配置

```
dwc_control_usb: dwc-control-usb {
    compatible = "rockchip,rk322xh-dwc-control-usb";
    rockchip,grf = <&usb2phy_grf>;
    interrupts = <GIC_SPI 59 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 60 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 61 IRQ_TYPE_LEVEL_HIGH>,
                <GIC_SPI 62 IRQ_TYPE_LEVEL_HIGH>;
    interrupt-names = "otg_bvalid", "otg_id",
                    "otg_linestate", "host0_linestate";
    status = "disabled";
    usb_bc {
        compatible = "inno,phy";
        regbase = &dwc_control_usb;
        rk_usb,bvalid = <0x120 9 1>;
        rk_usb,iddig = <0x120 6 1>;
        rk_usb,vdmsrcen = <0x108 12 1>;
        rk_usb,vdpsrcen = <0x108 11 1>;
        rk_usb,rdmpden = <0x108 10 1>;
        rk_usb,idpsrcen = <0x108 9 1>;
        rk_usb,idmsinken = <0x108 8 1>;
        rk_usb,idpsinken = <0x108 7 1>;
        rk_usb,dpattach = <0x120 25 1>;
        rk_usb,cpdet = <0x120 24 1>;
        rk_usb,dcppattach = <0x120 23 1>;
    };
};
```

图表 4-9 USB 2.0 OTG PHY 节点的 DTSI 配置

### 4.2.3 USB 3.0 HOST Controller DTS

USB3.0 HOST 控制器为 XHCI，集成于 DWC3 OTG IP 中，所以不用单独配置 dts，只需要配置 DWC3，并且设置 DWC3 的 dr\_mode 属性为 dr\_mode = "host"，即可以 enable XHCI 控制器。“phys”属性需要引用 USB 3.0 PHY 的 u3phy\_utmi 和 u3phy\_pipe 节点。

详细的配置说明，请参考文档：

Documentation/devicetree/bindings/usb/rockchip,dwc3.txt

```
usbdrd3: usb@ff600000 {
    compatible = "rockchip,rk322xh-dwc3";
    clocks = <&clk_usb3otg_ref>, <&clk_usb3otg0_s>,
            <&clk_gates19 14>;
    clock-names = "ref_clk", "suspend_clk",
                  "bus_clk";
    #address-cells = <2>;
    #size-cells = <2>;
    ranges;
    status = "disabled";
    usbdrd_dwc3: dwc3@ff600000 {
        compatible = "snps,dwc3";
        reg = <0x0 0xff600000 0x0 0x100000>;
        interrupts = <GIC_SPI 67 IRQ_TYPE_LEVEL_HIGH>;
        dr_mode = "host";
        phys = <&u3phy_utmi>, <&u3phy_pipe>;
        phy-names = "usb2-phy", "usb3-phy";
        phy_type = "utmi_wide";
        snps,dis_enblslpm_quirk;
        snps,dis-u2-freeclk-exists-quirk;
        snps,dis_u2_susphy_quirk;
        snps,dis-del-phy-power-chg-quirk;
        snps,dis-u3-autosuspend-quirk;
        status = "disabled";
    };
};
```

图表 4-10 USB 3.0 Host DTSI 配置

# 5 驱动开发

本章节主要对 USB 控制器和 PHY 的驱动框架以及驱动的调试接口作简要描述。

## 5.1 USB PHY drivers

USB PHY drivers 基于 Generic PHY Framework (Documentation/phy.txt)，代码位于 drivers/phy 目录下。

### 5.1.1 USB 2.0 PHY driver

1. Driver 代码路径: drivers/phy/phy-rockchip-usb.c

该 driver 的作用是为 USB 2.0 HOST 控制器提供控制 USB 2.0 PHY 的接口，以及做 USB 2.0 PHY 信号的 tuning。

2. 主要的函数说明如下：

rk3\*\_usb\_phy\_power: 控制不同 SoC USB 2.0 PHY 的 suspend/resume，以节省功耗；

rockchip\_usb\_phy\_power\_on/off: 提供给 USB 2.0 HOST 控制器 power on/off PHY 的接口；

rk3\*\_usb\_phy\_tuning: USB 2.0 PHY 信号的 tuning；

### 5.1.2 USB 3.0 PHY driver

1. Driver 代码路径: drivers/phy/phy-rockchip-inno-usb3.c

该 driver 的作用是为 USB 3.0 HOST 控制器提供控制 USB 3.0 PHY 的接口，以及做 USB 3.0 PHY 信号的 tuning。

2. 主要的函数说明如下：

rockchip\_u3phy\_usb2\_only\_\*: 用于 disable USB 3.0 PHY 的 super-speed；

rockchip\_u3phy\_rest\_\*: 用于复位 USB 3.0 PHY IP 内部的模块；

rockchip\_u3phy\_clk\_\*: 用于 USB 3.0 PHY clock 的控制；

rockchip\_u3phy\_power\_on/off: 提供 USB 3.0 HOST 控制器 power on/off PHY 的接口；

rockchip\_u3phy\_um\_sm\_work: 用于检测 USB 3.0 PHY utmi 的 line state 和 disconnect 状态；

rockchip\_u3phy\_port\_init: 用于 USB 3.0 PHY 硬件初始化；

rk322xh\_u3phy\_tuning: 用于 USB 3.0 PHY 的信号 tuning；

3. 主要的结构体：

static const struct rockchip\_u3phy\_cfg rk322xh\_u3phy\_cfgs[]

用于描述 USB 3.0 PHY 状态和控制寄存器；

4. 内核的调试接口：

USB 3.0 PHY 驱动提供了一个 u3phy\_mode 节点，用于 enable/disable USB 3.0 PHY 的 super-speed，路径在 /sys/kernel/debug/ff470000.usb3-phy/u3phy\_mode

使用方法如下：

1. Config to usb3.0 mode (enable super-speed)  
echo u3 > /sys/kernel/debug/ff470000.usb3-phy/u3phy\_mode
2. Config to usb2.0 only mode (disable super-speed)  
echo u2 > /sys/kernel/debug/ff470000.usb3-phy/u3phy\_mode

## 5.2 USB Controller drivers

### 5.2.1 USB 2.0 HOST Controller driver

Driver 代码路径: drivers/usb/host/ehci-\*.c  
drivers/usb/host/ohci-\*.c

其中, 板级相关的 platform 文件为:

ehci-platform.c  
ohci-platform.c

Rockchip EHCI&OHCI 控制器为标准的控制器设计, 采用 kernel 标准的驱动, 具体驱动的框架说明, 可以上网查阅。

### 5.2.2 USB 2.0 OTG Controller driver

1. Driver 代码路径: drivers/usb/dwc\_otg\_310/

2. 主要文件功能描述如下:

usbdev\_rk322xh.c : USB2.0 OTG 相关 DTS 解析与配置, 主要包括 clock、vbus gpio、linestate/bvalid/id 中断管理及 OTG PHY 相关的 GRF 配置;

dwc\_otg\_driver.c : 该文件为 controller 驱动入口, 主要包括 controller Register Config、Controller probe、Controller Supsend/Resume、Controller Mode Switch 等功能;

dwc\_otg\_pcd\_\*.c : 该系列文件为 Controller Peripheral 模式驱动功能;

dwc\_otg\_hcd\_\*.c : 该系列文件为 Controller Host 模式驱动功能;

usbdev\_bc.c: 实现充电检测功能;

3. 内核的调试接口:

#### 3.1 控制器寄存器 dump

调试接口位于/sys/devices/ff580000.usb 路径下:

执行 cat /sys/devices/ff580000.usb/regdump 打印 OTG 所有寄存器的状态

#### 3.2 控制器 device/host 强制切换

USB2.0 OTG 控制器的模式一般由 USB ID 电平决定, 也可以由软件进行强制切换。

调试接口为/sys/devices/ff580000.usb/driver/force\_usb\_mode

force\_usb\_mode 可能的值为:

0: force to OTG

1: force to Host

2: force to Peripheral

除此, Peripherals 模式也可以通过 UI 界面进行设置, 具体是: 勾选 Android 系统设置→设备→USB→连接到 PC 来设置。

### 5.2.3 USB 3.0 HOST Controller driver

1. Driver 代码路径: drivers/usb/dwc3/  
driver/usb/host/xhci-\*.c

2. 主要文件功能描述如下:

drivers/usb/dwc3/dwc3-rk322xh.c: rk3328/rk322xh platform 驱动;

drivers/usb/dwc3/core.c: DesignWare USB3 DRD Controller Core file,实现 dwc3 寄存器的初始化;

drivers/usb/dwc3/gadget.c: 实现 dwc3 peripheral 模式的驱动功能;

drivers/usb/dwc3/host.c: 实现 dwc3 host 模式的驱动功能;

drivers/usb/host/xhci-plat.c: xHCI host controller driver platform Bus Glue;

3. 内核的调试接口:

#### 3.1. DWC3 控制器寄存器 dump

执行命令 `cat /sys/kernel/debug/ff600000.dwc3/regdump`, 打印 DWC3 控制器的所有寄存器状态

#### 3.2 设置控制器进入 compliance mode, 用于 USB 2.0/3.0 信号质量测试

调试接口为 `/sys/kernel/debug/usb.24/host_testmode`

使用方法:

1. set test packet for the USB2 port of USB3 interface:

```
echo test_packet > /sys/kernel/debug/usb.23/host_testmode
```

2. set compliance mode for the USB3 port of USB3 interface:

```
echo test_u3 > /sys/kernel/debug/usb.23/host_testmode
```

3. check the testmode status:

```
cat /sys/kernel/debug/usb.23/host_testmode
```

The log maybe like this:

```
U2: test_packet /* means that U2 in test mode */
```

```
U3: compliance mode /* means that U3 in test mode */
```

# 6 Android Gadget 配置

## 6.1 Gadget 驱动配置

请参阅 3.4 USB Gadget 配置 章节。

## 6.2 BOOT IMG 配置

在 Android boot.img 中与 USB 相关的 script 主要有：

```
init.usb.rc,  
init.rk30board.usb.rc  
init.usbstorage.rc
```

- 1) init.usb.rc 为 Android 标准 rc 文件，一般不需要改动。
- 2) init.rk30board.usb.rc 为 Rockchip 平台 Gadget 功能的配置管理文件，其内容主要包括 usb gadget functiont 描述符的定义（VID/PID）、function 节点的使能等，如下所示为该文件的部分截图：

```
on init  
# write /sys/class/android_usb/android0/iSerial ${ro.serialno}  
# write /sys/class/android_usb/android0/f_rndis/manufacturer RockChip  
# write /sys/class/android_usb/android0/f_rndis/vendorID 2207  
# write /sys/class/android_usb/android0/f_rndis/wceis 1  
  
on boot  
write /sys/class/android_usb/android0/iSerial ${ro.serialno}  
write /sys/class/android_usb/android0/f_rndis/manufacturer RockChip  
write /sys/class/android_usb/android0/f_rndis/vendorID 2207  
write /sys/class/android_usb/android0/f_rndis/wceis 1  
write /sys/class/android_usb/android0/iManufacturer ${ro.product.manufacturer}  
write /sys/class/android_usb/android0/iProduct ${ro.product.model}  
write /sys/class/android_usb/android0/f_mass_storage/inquiry_string $ro.product.usbfactory  
  
on fs  
mkdir /dev/usb-ffs 0770 shell shell  
mkdir /dev/usb-ffs/adb 0770 shell shell  
mount functionfs adb /dev/usb-ffs/adb uid=2000,gid=2000  
write /sys/class/android_usb/android0/f_ffs/aliases adb  
  
on property:sys.usb.config=adb  
write /sys/class/android_usb/android0/enable 0  
write /sys/class/android_usb/android0/idVendor 2207  
write /sys/class/android_usb/android0/idProduct 0006  
write /sys/class/android_usb/android0/functions ${sys.usb.config}  
write /sys/class/android_usb/android0/enable 1  
start adbd  
setprop sys.usb.state ${sys.usb.config}
```

```
on property:sys.usb.config=mtp
    write /sys/class/android_usb/android0/enable 0
    write /sys/class/android_usb/android0/idVendor 2207
    write /sys/class/android_usb/android0/idProduct 0001
    write /sys/class/android_usb/android0/functions ${sys.usb.config}
    write /sys/class/android_usb/android0/enable 1
    setprop sys.usb.state ${sys.usb.config}

on property:sys.usb.config=mtp,adb
    write /sys/class/android_usb/android0/enable 0
    write /sys/class/android_usb/android0/idVendor 2207
    write /sys/class/android_usb/android0/idProduct 0011
    write /sys/class/android_usb/android0/functions ${sys.usb.config}
    write /sys/class/android_usb/android0/enable 1
    start abdd
    setprop sys.usb.state ${sys.usb.config}

on property:sys.usb.config=rndis
    write /sys/class/android_usb/android0/enable 0
    write /sys/class/android_usb/android0/idVendor 2207
    write /sys/class/android_usb/android0/idProduct 0003
    write /sys/class/android_usb/android0/functions ${sys.usb.config}
    write /sys/class/android_usb/android0/bDeviceClass 224
    write /sys/class/android_usb/android0/enable 1
    setprop sys.usb.state ${sys.usb.config}
```

图表 6-1 Android Gadget init.rk30board.usb.rc 的 usb 配置信息

on init/on boot 节点为 Android usb 描述符信息。其中，iSerial、iManufacturer、iProduct 三个属性由 Android 配置。如果 iSerial 没有配置成功，可能会造成 ADB 无法使用。

on property 节点为 setprop 提供配置，主要用于 gadget composite function 的切换。目前 RK3328 SDK 主要支持的 function 主要包括如下几种。

```
adb;
mtp;
adb,mtp;
rndis;
rndis,adb;
ptp;
ptp,adb;
mass_storage;
mass_storage,adb;
accessory;
accessory,adb;
acm;
acm,adb;
```



# 7 常见问题分析

## 7.1 设备枚举日志

### 7.1.1 USB 2.0 OTG 正常开机日志

```
开机未连线，默认为 device 模式
[9.731330] [0: kworker/0:1: 30] [otg id chg] last id -1 current id 64
[9.731446] [0: kworker/0:1: 30] PortPower off
[9.731489] [0: kworker/0:1: 30] rk_battery_charger_detect_cb,
battery_charger_detect 6
[9.831065] [0: kworker/0:1: 30] Using Buffer DMA mode
[9.831097] [0: kworker/0:1: 30] Periodic Transfer Interrupt Enhancement - disabled
[9.831115] [0: kworker/0:1: 30] Multiprocessor Interrupt Enhancement - disabled
[9.831135] [0: kworker/0:1: 30] OTG VER PARAM: 0, OTG VER FLAG: 0
[9.831152] [0: kworker/0:1: 30] ^^^^^^^^^^^^^^^^^Device Mode
```

### 7.1.2 USB 2.0 OTG Device 正常连接日志

```
mtp+adb 模式
[16.245909][0: kworker/0:1: 30] *****vbus detect*****
[16.370776][0: kworker/0:1: 30] Using Buffer DMA mode
[16.370800][0: kworker/0:1: 30] Periodic Transfer Interrupt Enhancement - disabled
[16.370828][0: kworker/0:1: 30] Multiprocessor Interrupt Enhancement - disabled
[16.370847][0: kworker/0:1: 30] OTG VER PARAM: 0, OTG VER FLAG: 0
[16.370863][0: kworker/0:1: 30] ^^^^^^^^^^^^^^^^^Device Mode
[16.370931][0: kworker/0:1: 30] *****soft connect!!!*****
[16.479209][0: swapper/0: 0] USB RESET
[16.574008][0: kworker/0:1: 30] android_work: sent uevent
USB_STATE=CONNECTED
[16.576041][0: swapper/0: 0] USB RESET
[16.706051][0: swapper/0: 0] android_usb gadget: high-speed config #1:
android
[16.706356][0: kworker/0:1: 30] android_work: sent uevent
USB_STATE=CONFIGURED
[16.733446][3:d.process.media: 735] mtp_open
```

### 7.1.3 USB 2.0 OTG Device 正常断开日志

```
[22.817708][0: swapper/0: 0] *****session end ,soft disconnect*****
[22.818011][0: kworker/0:1: 30] android_work: sent uevent
USB_STATE=DISCONNECTED
[22.818062][0: kworker/0:1: 30] android_work: did not send uevent (0 0
(null))
[22.818319][0: MtpServer: 922] mtp_release
```

### 7.1.4 USB 2.0 OTG HOST 设备正常连接日志

#### LS 设备

```
[71.985341][2: khubd: 40] usb 5-1: new low-speed USB device number 2 using
usb20_otg
[71.986121][0: khubd: 40] Indeed it is in host mode hprt0 = 00041901
[72.166594][0: khubd: 40] usb 5-1: New USB device found, idVendor=046d,
idProduct=c077
[72.166655][0: khubd: 40] usb 5-1: New USB device strings: Mfr=1, Product=2,
SerialNumber=0
[72.166706][0: khubd: 40] usb 5-1: Product: USB Optical Mouse
[72.166752][0: khubd: 40] usb 5-1: Manufacturer: Logitech
[72.175046][0: khubd: 40] input: Logitech USB Optical Mouse as
/devices/ff580000.usb/usb5/5-1/5-1:1.0/input/input2
```

#### FS 设备

```
[40.452561][3: khubd: 40] usb 5-1: new full-speed USB device number 2 using
usb20_otg
[40.632926][0: khubd: 40] usb 5-1: New USB device found, idVendor=1915,
idProduct=0199
[40.632993][0: khubd: 40] usb 5-1: New USB device strings: Mfr=1, Product=2,
SerialNumber=0
[40.633042][0: khubd: 40] usb 5-1: Product: Memsart controller
[40.633088][0: khubd: 40] usb 5-1: Manufacturer: Memsart
[40.644143][0: khubd: 40] input: Memsart Memsart controller as
/devices/ff580000.usb/usb5/5-1/5-1:1.0/input/input2
```

#### HS 设备

```
[26.943532][2: khubd: 40] usb 5-1: new high-speed USB device number 2 using
usb20_otg
[26.943885][0: khubd: 40] Indeed it is in host mode hprt0 = 00001101
[27.055019][0: khubd: 40] Indeed it is in host mode hprt0 = 00001501
[27.383456][0: khubd: 40] usb 5-1: New USB device found, idVendor=0951,
idProduct=1687
[27.383520][0: khubd: 40] usb 5-1: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[27.383570][0: khubd: 40] usb 5-1: Product: DT R400
[27.383614][0: khubd: 40] usb 5-1: Manufacturer: Kingston
```

### 7.1.5 USB 2.0 HOST 设备正常连接日志

#### LS 设备

```
[38.707972] [3: khubd: 40] usb 4-1: new low-speed USB device number 2 using
ohci-platform
[38.895308] [0: khubd: 40] usb 4-1: New USB device found, idVendor=03f0,
idProduct=2c24
[38.895369] [0: khubd: 40] usb 4-1: New USB device strings: Mfr=1, Product=2,
```

```

SerialNumber=0
[38.895422] [0: khubd: 40] usb 4-1: Product: HP USB Laser Mouse
[38.895467] [0: khubd: 40] usb 4-1: Manufacturer: HP
[38.907013] [0: khubd: 40] input: HP HP USB Laser Mouse as
/devices/ff5d0000.usb/usb4/4-1/4-1:1.0/input/input2
[38.909237] [0: khubd: 40] hid-generic 0003:03F0:2C24.0001: input,hidraw0:
USB HID v1.10 Mouse [HP HP USB Laser Mouse] on usb-ff5d0000.usb-1/input0

```

#### FS 设备

```

[1: khubd: 40] usb 4-1: new full-speed USB device number 3 using ohci-platform
[79.655165] [0: khubd: 40] usb 4-1: New USB device found, idVendor=045e,
idProduct=07b2
[79.655225] [0: khubd: 40] usb 4-1: New USB device strings: Mfr=1, Product=2,
SerialNumber=0
[79.655276] [0: khubd: 40] usb 4-1: Product: Microsoft® Nano Transceiver v1.0
[79.655323] [0: khubd: 40] usb 4-1: Manufacturer: Microsoft
[79.676566] [0: khubd: 40] input: Microsoft Microsoft® Nano Transceiver v1.0 as
/devices/ff5d0000.usb/usb4/4-1/4-1:1.0/input/input3

```

#### HS 设备

```

[3: khubd: 40] usb 3-1: new high-speed USB device number 3 using ehci-platform
[ 80.957315] [0: khubd: 40] usb 3-1: New USB device found, idVendor=0930,
idProduct=6544
[ 80.957402] [0: khubd: 40] usb 3-1: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[ 80.957452] [0: khubd: 40] usb 3-1: Product: TransMemory
[ 80.957496] [0: khubd: 40] usb 3-1: Manufacturer: TOSHIBA
[ 80.957541] [0: khubd: 40] usb 3-1: SerialNumber:
322EA08DAD86CF111837E2EC
[ 80.959008] [0: khubd: 40] usb-storage 3-1:1.0: USB Mass Storage device
detected
[ 80.960465] [0: khubd: 40] scsi0 : usb-storage 3-1:1.0
...
[ 81.085812] [3: kworker/u8:4: 913] sd 0:0:0:0: [sda] Attached SCSI removable
disk

```

### 7.1.6 USB 2.0 HOST-LS/FS/HS 设备断开日志

```
[ 443.151067] usb 4-1: USB disconnect, device number 3
```

### 7.1.7 USB 3.0 HOST-SS 设备正常连接日志

```

[22.019722] [0: khubd:40] usb 2-1: new SuperSpeed USB device number 2 using
xhci-hcd
[22.033189] [0: khubd:40] usb 2-1: Parent hub missing LPM exit latency info.
Power management will be impacted.
[22.034154] [0: khubd:40] usb 2-1: New USB device found, idVendor=0781,
idProduct=5581

```

```
[22.034207] [0: khubd:40] usb 2-1: New USB device strings: Mfr=1, Product=2,
SerialNumber=3
[22.034258] [0:khubd:40] usb 2-1: Product: SanDisk Ultra
[22.034306] [0:khubd:40] usb 2-1: Manufacturer: SanDisk
[22.034350] [0:khubd:40] usb 2-1: SerialNumber: A2004BD1F8067C73
[22.039137] [0:khubd:40] usb-storage 2-1:1.0: USB Mass Storage device detected
[22.039896] [0:khubd:40] scsi2 : usb-storage 2-1:1.0
[23.027377] [1:kworker/u8:4:923] scsi 2:0:0:0: Direct-Access      SanDisk
SanDisk Ultra      PMAP PQ: 0 ANSI: 6
```

## 7.2 USB 常见问题分析

### 7.2.1 软件配置

必须明确项目中 USB 控制器是如何分配的,并确保 kernel 的配置是正确的,请参考第 3、4 章的配置说明,根据项目的实际使用情况进行配置。

### 7.2.2 硬件电路

在同时使用多个控制器对应同一个 USB 口,或者一个控制器对应多个 USB 口时,可能会使用电子开关来切换 USB 信号及电源。需要确保不同控制器的电源控制是互相独立的,通过电子开关后,控制器与 USB 口之间的连接是有效的。

场景一:

1 个硬件 USB 口同时支持 HOST 和 device 功能,使用 USB2.0 HOST 控制器作为 HOST 和 USB2.0 OTG 控制器作为 device,通过硬件电子开关进行切换。

需要保证工作于 HOST 状态时,USB 信号是切换到 USB2.0 HOST 控制器,而 VBUS 是由 HOST 供电电路提供,而不影响 device 的 VBUS 电平检测电路。工作于 device 状态时,USB 信号是切换到 USB2.0 OTG 控制器,VBUS 由 PC 通过 USB 线提供。

场景二:

使用一个 USB2.0 OTG 控制器,对应使用两个硬件 USB 口分别是 HOST 和 Device。通过电子开关进行信号切换。

工作于 HOST 状态时,USB2.0 OTG 的 DP/DM 信号线是切换到 HOST 口,且 HOST 口 VBUS 提供 5V 500MA 的供电;工作于 device 状态时 DP/DM 信号是切换到 device 口,VBUS 电平检测电路只检测 PC 提供的 5V 供电。

### 7.2.3 Device 功能异常分析

**USB Device 正常连接至 PC 的现象主要有:**

1. 串口输出正常 log 见 [7.1.2 USB 2.0 OTG Device 正常连接日志](#);
2. PC 出现盘符,但默认不能访问;(windows 7 和 MAC OS 可能只出现在设备管理器);
3. 设备 UI 状态栏出现“USB 已连接”标识;
4. 打开 USB 已连接的提示窗口,默认为 charger only 模式,选择“MTP”或者“PTP”后,PC 可以访问盘符。

**常见异常排查:**

**1、连接 USB 时串口完全没有 log:**

- (1) USB 硬件信号连接正确;
- (2) USB 控制器确保工作在 device 状态;
- (3) 测量 USB\_DET 信号电压,USB 连接时应该由低到高。

**2、连接失败，PC 显示不可识别设备，log 一直重复打印：**

```
[36.682587] DWC_OTG: *****soft
connect!!!*****
[36.688603] DWC_OTG: USB SUSPEND
[36.807373] DWC_OTG: USB RESET
```

但是没有正常 log 中的后面几条信息。

一般为 USB 硬件信号差，无法完成枚举。

**3、连接 PC 后，kernel log 正常，并且设备为出现“USB 已连接”标识，但 PC 无法访问设备**

驱动工作正常，请先确认是否有选择 USB 为“MTP”或“PTP”，如果已选择，则可能是 android 层异常，请截取 logcat 内容，并请负责维护 vold/mtpserver 代码的 android 工程师帮忙 debug。

**4、连接 PC 正常，并能正常访问，拷贝文件过程中提示拷贝失败。**

可能原因是：

(1) USB 信号质量差。可测试下 USB 眼图，并使用 USB 分析仪抓取数据流后分析。

(2) flash/sd 卡读写超时，log 一般为连接 window xp 时约 10S 出现一次重新连接的 log。

(3) flash/sd 磁盘分区出错，导致每次拷贝到同一个点时失败。可使用命令检查并修复磁盘分区。假设挂载的磁盘分区为 E，则打开 windows 命令提示符窗口，输入命令：**chkdsk E: /f**

**5、USB 线拔掉后 UI 状态栏仍然显示“USB 已连接”，或 USB 线拔掉时只有以下 log：**

```
[25.330017] DWC_OTG: USB SUSPEND
而下面的 log:
[25.514407] DWC_OTG: *****session end intr, soft
disconnect*****
```

VBUS 异常，一直为高，会影响 USB 检测及系统休眠唤醒，请硬件工程师排查问题。

## 7.2.4 Host 功能异常分析

USB HOST 正常工作情况如下：

1. 首先 HOST 电路提供 5V，至少 500mA 的供电；
2. 如果有 USB 设备连接进来，串口首先会打印 HOST 枚举 USB 设备的 log(见 [7.1.4](#) 和 [7.1.5](#))，表明 USB 设备已经通过 HOST 的标准设备枚举；

**常见异常及排查：**

**1. HOST 口接入设备后，串口无任何打印：**

- (1) 首先需要确认通过电子开关后的电路连接正确；
- (2) 确认控制器工作于 HOST 状态，并确认供电电路正常。

**2. 串口有 HOST 枚举 USB 设备内容，但是没有出现 class 驱动的打印信息。**

Kernel 没有加载 class 驱动，需要重新配置 kernel，加入对应 class 驱动支持。

**3. kernel 打印信息完整(USB 标准枚举信息及 CLASS 驱动信息)，已在 Linux 对应位置生成节点，但是 android 层无法使用。**

Android 层支持不完善，如 U 盘在 kernel 挂载完成/dev/block/sda 节点后，需要 android 层 vold 程序将可存储介质挂载到/udisk 提供媒体库，资源管理器等访问，同样鼠标键盘等 HID 设备也需要 android 层程序支持。

U 盘枚举出现/dev/block/sda 后仍然无法使用，一般是 fstab.rk30board 中 U 盘的 mount 路径有问题，fstab.rk30board 的代码如下(系统起来后可直接 cat fstab.rk30board 查看)：

```
/devices/ff5c0000.usb /mnt/usb_storage/USB_DISK0 vfat defaults
```

```
voldmanaged=usb_storage:auto
    /devices/ff5d0000.usb    /mnt/usb_storage/USB_DISK1    vfat    defaults
voldmanaged=usb_storage:auto
    /devices/ff580000.usb    /mnt/usb_storage/USB_DISK2    vfat    defaults
voldmanaged=usb_storage:auto
    /devices/usb.            /mnt/usb_storage/USB_DISK3    vfat    defaults
voldmanaged=usb_storage:auto
```

而实际的 device 路径可能改变, 与 fstab.rk30board 中的配置不一致。如果设备属于这种情况的无法正常使用, 需要联系 android 工程师帮忙 debug。

#### 4. OTG 口作为 host 时, 无法识别接入的设备

- (1) 检查 kernel 的 OTG 配置是否正确;
- (2) 检查 OTG 电路的 ID 电平(作 host, 为低电平)和 VBUS 5V 供电是否正常;
- (3) 如果确认 1 和 2 都正常, 仍无法识别设备, 请提供设备插入后无法识别的错误 log 给我们。

### 7.2.5 USB Camera 异常分析

#### 1. 使用 Camera 应用, 无法打开 USB camera

首先, 检查/dev 目录下是否存在 camera 设备节点 video0 或 video1, 如果不存在, 请检查 kernel 的配置是否正确, 如果存在节点, 请确认 USB camera 是在系统开机前插入的, 因为 RK 平台的 SDK, 默认是不支持 USB camera 热拔插的。如果要支持 USB camera 热拔插, 请联系负责 camera 的工程师修改 Android 相关代码, USB 驱动不需要做修改。

如果仍无法解决, 请提供 log 给负责 USB 驱动工程师或者负责 camera 的工程师, 进一步分析。

#### 2. 出现概率性闪屏、无图像以及 camera 应用异常退出的问题

可能是 USB 驱动丢帧导致的。需要使用 USB 分析仪抓实际通信的数据进行分析, 如果无法定位, 请联系负责 USB 驱动的工程师。

### 7.2.6 USB 充电检测

RK3328 USB2 PHY 支持 BC1.2 标准的充电检测, 代码实现请参考 drivers/phy/phy-rockchip-inno-usb2.c, 可以检测 SDP/CDP/标准 DCP(D+/D-短接)/非标准 DCP(D+/D-未短接)四种充电类型。

#### ● SDP —— Standard Downstream Port

根据 USB2.0 规范, 当 USB 外设处于未连接(un-connect)或休眠(suspend)的状态时, 一个 Standard Downstream Port 可向该外设提供不超过 2.5mA 的平均电流;当外设处于已经连接并且未休眠的状态时, 电流可以至最大 100mA(USB3.0 150mA);而当外设已经配置(configured)并且未休眠时, 最大可从 VBUS 获得 500mA(USB3.0 900mA)电流。

#### ● CDP —— Charging Downstream Port

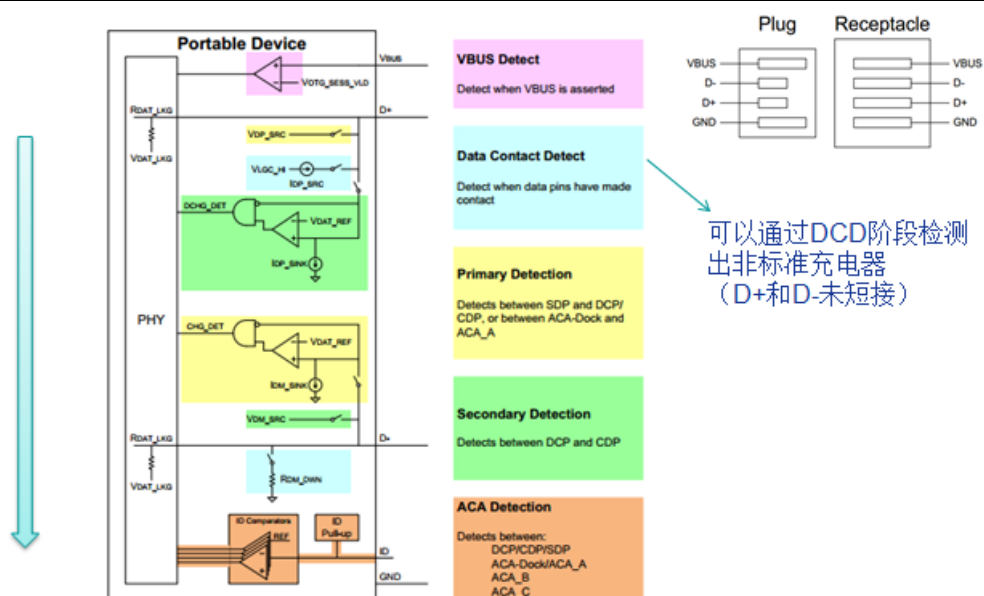
即兼容 USB2.0 规范, 又针对 USB 充电作出了优化的下行 USB 接口, 提供最大 1.5A 的供电电流, 满足大电流快速充电的需求。

#### ● DCP —— Dedicated Charging Port (USB Charger)

BC1.2 spec 要求将 USB Charger 中的 D+和 D-进行短接, 以配合 USB 外设的识别动作, 但它不具备和 USB 设备通信的能力。

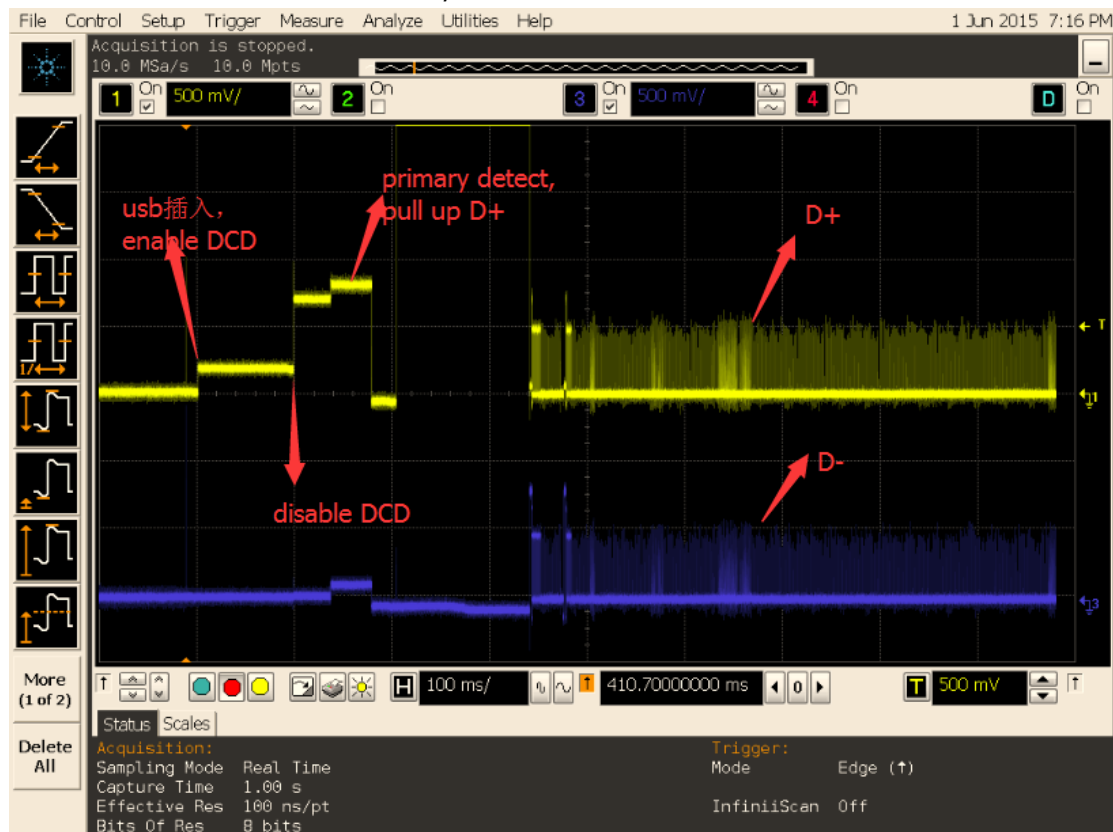
USB 充电类型检测流程见下图所示:





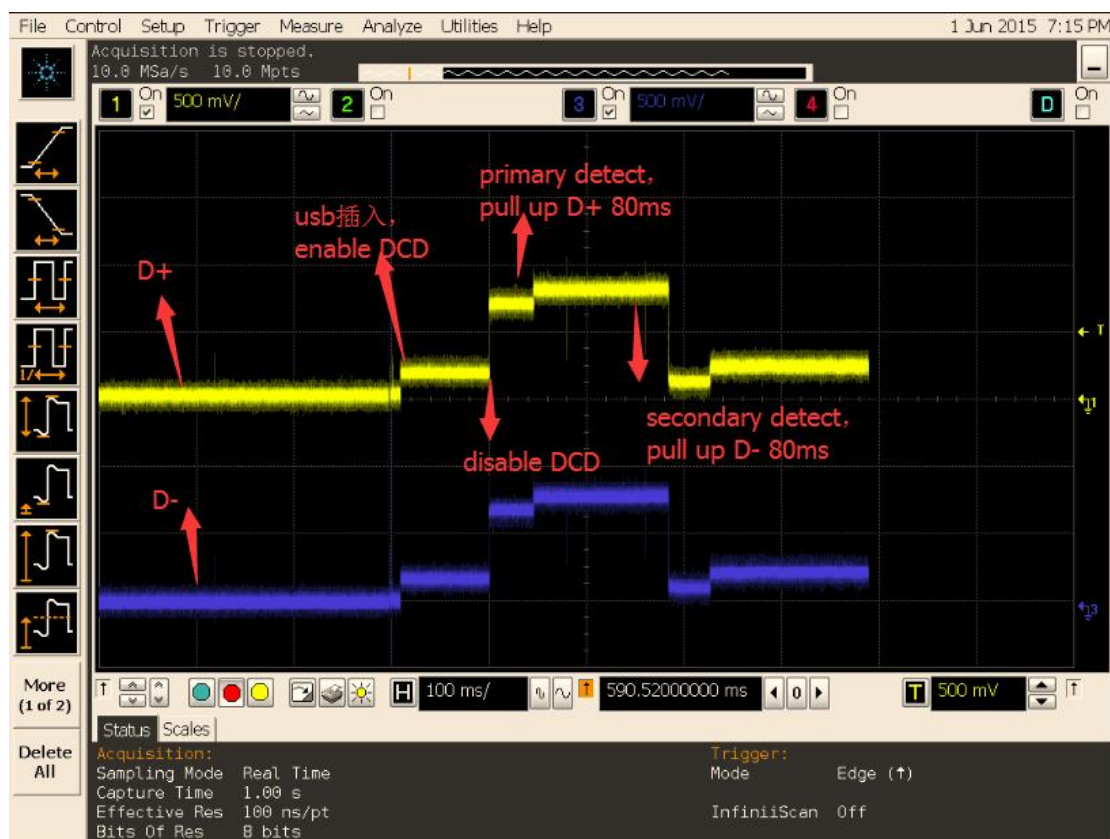
图表 7-1 USB 充电检测流程

典型的 SDP 检测过程中，D+/D- 波形如下图所示：



图表 7-2 SDP 检测波形

典型的 DCP 检测过程中，D+/D- 波形如下图所示：



图表 7-3 DCP 检测波形

如果连接 USB 充电器，发现充电慢，有可能是 DCP 被误检测为 SDP，导致充电电流被设置为 500mA。当 USB 线连接不稳定或者充电检测驱动出错，都可能会产生该问题。解决方法：

抓取 USB 充电器连接的 log，通过 log 的提示判断检测的充电类型，正常应为 DCP；

如果连接的是 USB 充电器，但 log 提示为 SDP，则表示发生了误检测。请先更换 USB 线测试，并使用万用表确认 D+/D- 是否短接。如果仍无法解决，请将检测的 log 发给我们测试。同时，如果有条件，请使用示波器抓 USB 插入时的 D+/D- 波形，并连同 log 一起发送给我们分析和定位问题。

如果连接的是 USB 充电器，并且 log 提示为 DCP，但充电仍然很慢，则表明软件检测正常，可能是充电 IC 或者电池的问题。

## 7.3 PC 驱动问题

所有 USB 设备要在 PC 上正常工作都是需要驱动的，有些驱动是标准且通用的，而有些驱动是需要额外安装的。对于 RK 的设备连接到 PC 后，需要安装驱动的情况有两种的设备，需要分别选择对应的驱动。

1. 生成后未烧写的裸片或者进入升级模式后的 RK 设备，会以 rockUSB 的模式连接到 PC，需要在 PC 端使用 RK 平台专门的驱动安装助手 DriverAssitant（RK3328 需要 v4.4 支持）安装驱动才能识别到 USB 设备；
2. RK 的设备正常运行时，在设置里面打开了 USB debugging 选项，连接时会以 ADB 的模式连接 PC，同样需要在 PC 端使用 RK 平台专门的驱动安装助手 DriverAssitant 安装 ADB 驱动后，才能正常识别到 ADB 设备。



# 8 USB 信号测试

USB2.0/3.0 信号测试方法及常见问题分析请参阅《RK USB Compliance Test Note V1.2》