



RTOS USB 开发指南

**版本号: 1.1
发布日期: 2021.4.10**

版本历史

版本号	日期	制/修订人	内容描述
1.0	2020.7.7	Allwinner	1. 初版
1.1	2021.4.10	AWA1736	1. 完善 F133



目 录

1 前言	1
1.1 文档简介	1
1.2 目标读者	1
1.3 适用范围	1
2 模块介绍	2
2.1 模块功能介绍	2
2.2 相关术语介绍	2
2.2.1 硬件术语	2
2.2.2 软件术语	2
2.3 模块配置介绍	3
2.3.1 platform 配置说明	3
2.3.2 kernel menuconfig 配置说明	4
2.4 源码结构介绍	4
3 模块接口说明	8
3.1 HOST API	8
3.1.1 hal_usb_core_init	8
3.1.2 hal_usb_core_exit	8
3.1.3 hal_usb_hci_init	9
3.1.4 hal_usb_hcd_init	9
3.1.5 hal_usb_hcd_deinit	9
3.2 MANAGER API	9
3.2.1 hal_usb_manager_init	10
3.2.2 hal_usb_manager_deinit	10
3.3 UDC API	10
3.3.1 hal_udc_init	11
3.3.2 hal_udc_device_desc_init	11
3.3.3 hal_udc_config_desc_init	11
3.3.4 hal_udc_string_desc_init	11
3.3.5 hal_udc_register_callback	12
3.3.6 hal_udc_ep_read	12
3.3.7 hal_udc_ep_write	12
3.3.8 hal_udc_ep_enable	12
3.3.9 hal_udc_ep_set_buf	13
4 模块使用范例	14
5 FAQ	15
5.1 调试方法	15
5.1.1 调试工具	15
5.1.2 调试节点	15

5.2 常见问题	15
----------------	----



1 前言

1.1 文档简介

介绍 RTOS 中 USB 驱动接口及使用方法。

1.2 目标读者

USB 驱动、及应用层的开发/维护人员。

1.3 适用范围

表 1-1: 适用产品列表

产品名称	内核版本	驱动文件
F133	Melis	hal_udc.c
V459	Melis	hal_udc.c
R328	FreeRTOS	hal_udc.c
R329-DSP	FreeRTOS	hal_udc.c

2 模块介绍

2.1 模块功能介绍

BSP USB 驱动主要实现设备驱动的底层细节，并为上层提供一套标准的 API 接口以供使用。

2.2 相关术语介绍

2.2.1 硬件术语

术语	解释说明
EP	EndPoint, 端点
OTG	On-The-Go
HCD	Host Controller Driver, 主机控制器驱动
UDC	USB Device Controller, USB 设备控制器
HCI	Host Controller Interface, 主机控制器接口
EHCI	Enhanced Host Controller Interface, 增强型主机控制器接口
OHCI	Open Host Controller Interface, 开放式主机控制器接口

2.2.2 软件术语

术语	解释说明
HAL	Hardware Abstraction Layer, 硬件抽象层
RTOS	Real Time Operating System, 实时操作系统

2.3 模块配置介绍

2.3.1 platform 配置说明

不同的 Sunxi 硬件平台中配置，通过路径：rtos-hal/hal/source/usb/platform/sunxxx 下的两个文件来具体配置。具体配置方法如下所示：

- usb_sunxxx.h

```
//配置控制器个数
#define USB_MAX_CONTROLLER_COUNT      2

//配置基地址
#define SUNXI_USB_OTG_PBASE           0x04100000
#define SUNXI_USB_EHCI0_PBASE        0x04101000
#define SUNXI_USB_EHCI1_PBASE        0x04200000
```

- usb_sunxxx.c

```
//配置主机控制器
static const struct platform_usb_config usb_hci_table[] =
{
    { //ehci-0
        .name      = "sunxi-ehci0",
        .pbase     = SUNXI_USB_EHCI0_PBASE,
        .irq       = 62 - USB_RISCV_GIC_OFFSET, //注意中断号有offset
        .usb_clk   = CLK_BUS_EHCI0,
        .usb_rst   = RST_BUS_EHCI0,
        .phy_clk   = 0,
        .phy_rst   = RST_USB_PHY0
    },

    { //ehci-1
        .name      = "sunxi-ehci1",
        .pbase     = SUNXI_USB_EHCI1_PBASE,
        .irq       = 65 - USB_RISCV_GIC_OFFSET, //注意中断号有offset
        .usb_clk   = CLK_BUS_EHCI1,
        .usb_rst   = RST_BUS_EHCI1,
        .phy_clk   = 0,
        .phy_rst   = RST_USB_PHY1
    }
};

//配置otg
static const struct platform_usb_config usb_otg_table =
{
    .name      = "sunxi-otg",
    .pbase     = SUNXI_USB_OTG_PBASE,
    .irq       = 61 - USB_RISCV_GIC_OFFSET, //注意中断号有offset
    .usb_clk   = CLK_BUS_OTG,
    .usb_rst   = RST_BUS_OTG,
}
```

```
.phy_clk    = 0,
.phy_rst    = RST_USB_PHY0
};
```

说明

1. **SUNXI_USB_OTG_PBASE**, 表示 **UDC** 寄存器基地址;
2. **SUNXI_USB_EHCIX_PBASE**, 表示 **ehci** 寄存器基地址;

2.3.2 kernel menuconfig 配置说明

进入 lichee/melis-v3.0/目录下, 执行 `make menuconfig`, 进入配置主界面, 并按下图路径索引, 可配置 `usb` 的 `host/device` 功能, 以及功能驱动。

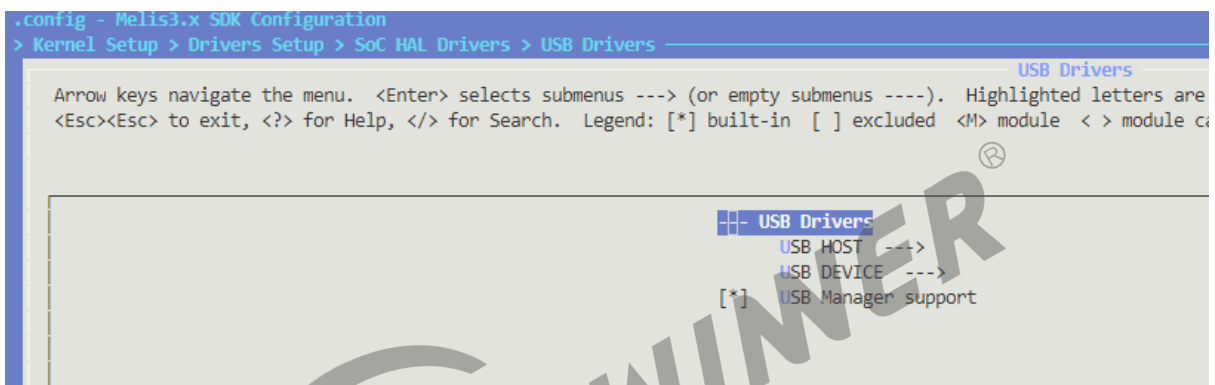


图 2-1: USB menuconfig 1

上述配置设置完毕后, 再次按照下图路径索引, 配置 `USB` 在 `RTOS` 系统中的加载项。

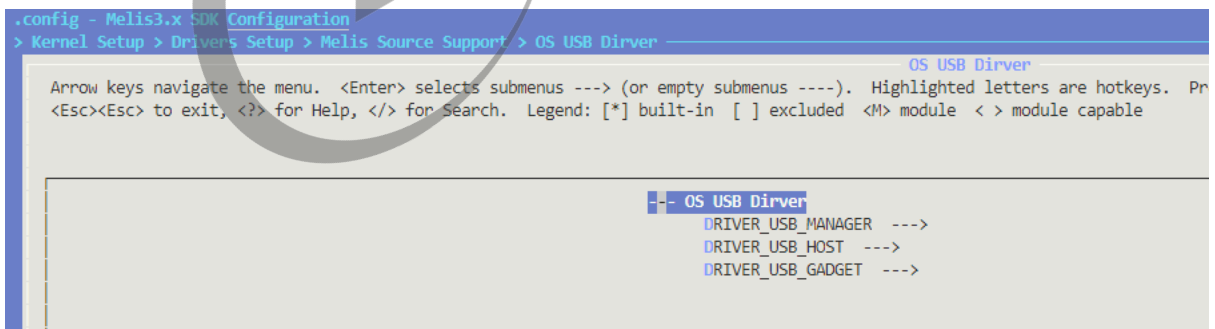


图 2-2: USB menuconfig 2

2.4 源码结构介绍

`RTOS` 中将 `usb` 驱动通用部分模块化, 搭配不同的 `RTOS` 系统来组成完整的 `USB` 功能驱动。因此代码分为两部分, `RTOS` 系统部分及 `RTOS-HAL` 部分。

- RTOS 系统部分

```
hal/source/usb/ ---- 驱动源码
├── core
│   └── core.c
├── gadget
│   ├── adb.c
│   ├── gadget.c
│   └── gadget.h
├── hci
│   └── hci.c
└── manager
    └── manager.c
```

- RTOS-HAL 部分

```
hal/source/usb/ ---- 驱动源码
├── core
│   ├── urb.c
│   ├── urb.h
│   ├── usb_core_base.c
│   ├── usb_core_base.h
│   ├── usb_core_config.c
│   ├── usb_core_config.h
│   ├── usb_core_init.c
│   ├── usb_core_init.h
│   ├── usb_core_interface.c
│   ├── usb_core_interface.h
│   ├── usb_driver_init.c
│   ├── usb_driver_init.h
│   ├── usb_gen_hci_rh.c
│   ├── usb_gen_hci_rh.h
│   ├── usb_gen_hci.c
│   ├── usb_gen_hci.h
│   ├── usb_gen_hub_base.c
│   ├── usb_gen_hub_base.h
│   ├── usb_gen_hub.c
│   ├── usb_gen_hub.h
│   ├── usb_msg_base.c
│   ├── usb_msg_base.h
│   ├── usb_msg.c
│   ├── usb_msg.h
│   ├── usb_virt_bus.c
│   └── usb_virt_bus.h
├── host
│   ├── ehci-hcd.c
│   ├── ehci-hub.c
│   ├── ehci-mem.c
│   ├── ehci-q.c
│   ├── ehci-sched.c
│   ├── ehci-sunxi.c
│   ├── ehci-timer.c
│   ├── ehci.h
│   ├── hal_hci.c
│   ├── sunxi-hci.c
│   └── sunxi-hci.h
└── manager
```

```
|   |— sunxi_usb_board.h
|   |— usb_hw_scan.c
|   |— usb_hw_scan.h
|   |— usb_manager_common.h
|   |— usb_manager.c
|   |— usb_msg_center.c
|   |— usb_msg_center.h
|— platform
|   |— sun8iw18
|   |   |— usb_sun8iw18.c
|   |   └─ usb_sun8iw18.c
|   |— sun8iw19
|   |   |— usb_sun8iw19.c
|   |   └─ usb_sun8iw19.c
|   └─ sun20iw1
|       |— usb_sun20iw1.c
|       └─ usb_sun20iw1.c
|— storage
|   |— Class
|   |   |— msc_common.h
|   |   |— mscProtocol.c
|   |   |— mscProtocol.h
|   |   |— mscTransport_i.c
|   |   |— mscTransport.c
|   |   |— mscTransport.h
|   |   └─ usb_msc.c
|   |— Disk
|   |   |— BlkDev.c
|   |   |— BlkDev.h
|   |   |— CD.c
|   |   |— CD.h
|   |   |— Disk.c
|   |   |— LunMgr_i.h
|   |   |— LunMgr.c
|   |   └─ Scsi2.c
|   |— Misc
|   |   |— usbh_buff_manager.c
|   |   |— usbh_disk_infi.c
|   |   └─ usbh_disk_remoce_time.c
|   └─ include
|       |— LunMgr.h
|       |— usb_msc_i.h
|       |— usb_msc.h
|       |— usb_buff_manager.h
|       |— usb_disk_info.h
|       |— usb_disk_remove_time.h
|       └─ Scsi2.h
|— include
|   |— ch9.h
|   |— ehci_def.h
|   |— error.h
|   |— list_head_ext.c
|   |— list_head_ext.h
|   |— mod_usbhost.h
|   |— platform_usb.h
|   |— platform_usbstorage.h
|   |— usb_drv_dev_match.c
|   |— usb_drv_dev_match.h
|   |— usb_gadget.h
|   └─ usb_gen_dev_mod.c
```

```
|— usb_gen_dev_mod.h
|— usb_host_common.h
|— usb_host_hub.h
|— usb_list.c
|— usb_list.h
|— usb_os_platform.c
|— usb_os_platform.h
|— usb_utils_find_zero_bit.c
|— usb_utils_find_zero_bit.h
```

include/hal/usb/ ---- 驱动APIs声明头文件

```
|— ch9.h
|— hal_hci.h
|— hal_manager.h
|— storage.h
|— sunxi_hal_udc.h
|— usb_gadget.h
```



3 模块接口说明

3.1 HOST API

API	解释说明
hal_usb_core_init	设置 usb core 初始化
hal_usb_core_exit	设置 usb core 退出
hal_usb_hci_init	加载所有（除了 0）主机驱动
hal_usb_hcd_init	加载具体 num 的主机驱动
hal_usb_hcd_deinit	卸载具体 num 的主机驱动

3.1.1 hal_usb_core_init

- 作用：设置 usb core 初始化
- 参数：
 - void：无返回
- 返回：
 - EPDK_OK：初始化成功
 - 其他：错误代码

3.1.2 hal_usb_core_exit

- 作用：设置 usb core 退出
- 参数：
 - void：无返回
- 返回：
 - EPDK_OK：初始化成功
 - 其他：错误代码

3.1.3 hal_usb_hci_init

- 作用：加载所有（除了 0）主机驱动
- 参数：
 - void：无返回
- 返回：
 - void：无返回

3.1.4 hal_usb_hcd_init

- 作用：加载具体 num 的主机驱动
- 参数：
 - hci_num：具体加载的驱动编号
- 返回：
 - -1：失败
 - 0：成功

3.1.5 hal_usb_hcd_deinit

- 作用：卸载具体 num 的主机驱动
- 参数：
 - hci_num：具体卸载的驱动编号
- 返回：
 - -1：失败
 - 0：成功

3.2 MANAGER API

API	解释说明
hal_usb_manager_init	manager 初始化
hal_usb_manager_deinit	manager 卸载

3.2.1 hal_usb_manager_init

- 作用：manager 初始化
- 参数：
 - void：无返回
- 返回：
 - -1：失败
 - 0：成功

3.2.2 hal_usb_manager_deinit

- 作用：manager 卸载
- 参数：
 - void：无返回
- 返回：
 - -1：失败
 - 0：成功

3.3 UDC API

API	解释说明
hal_udc_init	udc 初始化
hal_udc_deinit	udc 卸载 (预留)
hal_udc_enter_test_mode	进入 test 模式 (预留)
hal_udc_device_desc_init	设备描述符初始化
hal_udc_config_desc_init	配置描述符初始化
hal_udc_string_desc_init	字符串描述符初始化
hal_udc_register_callback	注册用户回调函数
hal_udc_ep_read	ep 读操作
hal_udc_ep_write	ep 写操作
hal_udc_ep_enable	使能 ep
hal_udc_ep_disable	关闭 ep (预留)
hal_udc_ep_set_buf	设置 ep 发送/接收 buffer

3.3.1 hal_udc_init

- 作用：UDC 初始化
- 参数：
 - void：无参数
- 返回：
 - void：无返回

3.3.2 hal_udc_device_desc_init

- 作用：设备描述符初始化
- 参数：
 - device_desc：设备描述符指针
- 返回：
 - void：无返回

3.3.3 hal_udc_config_desc_init

- 作用：配置描述符初始化
- 参数：
 - config_desc：配置描述符指针
 - len：配置描述符的长度
- 返回：
 - void：无返回

3.3.4 hal_udc_string_desc_init

- 作用：字符串描述符初始化
- 参数：
 - string_desc：字符串描述符指针
- 返回：
 - void：无返回

3.3.5 hal_udc_register_callback

- 作用：注册用户回调函数
- 参数：
 - user_callback: 回调函数指针
- 返回：
 - void: 无返回

3.3.6 hal_udc_ep_read

- 作用：ep 读操作
- 参数：
 - ep_addr: ep 地址
 - buf: 存放数据的 buf
 - len: 读的数据长度
- 返回：
 - 0: 读数据成功
 - 非 0: 错误代码

3.3.7 hal_udc_ep_write

- 作用：ep 写操作
- 参数：
 - ep_addr: ep 地址
 - buf: 存放数据的 buf
 - len: 写的数据长度
- 返回：
 - 0: 读数据成功
 - 非 0: 错误代码

3.3.8 hal_udc_ep_enable

- 作用：使能 ep
- 参数：

- ep_addr: ep 的地址
- maxpacket: 最大数据包
- ts_type: 传输类型
- 返回:
 - void: 无返回

3.3.9 hal_udc_ep_set_buf

- 作用: 设置 ep 发送/接收 buffer
- 参数:
 - ep_addr: ep 的地址
 - buf: 最大数据包
 - len: 传输类型
- 返回:
 - void: 无返回



4 模块使用范例

可参考驱动 APIs 测试代码（hal/test/usb/udc）。



5 FAQ

[介绍模块常见问题]

5.1 调试方法

5.1.1 调试工具

5.1.2 调试节点

5.2 常见问题






著作权声明

版权所有 © 2021 珠海全志科技股份有限公司。保留一切权利。

本档及内容受著作权法保护，其著作权由珠海全志科技股份有限公司（“全志”）拥有并保留一切权利。

本档是全志的原创作品和版权财产，未经全志书面许可，任何单位和个人不得擅自摘抄、复制、修改、发表或传播本档内容的部分或全部，且不得以任何形式传播。

商标声明

、 **全志科技** （不完全列举）均为珠海全志科技股份有限公司的商标或者注册商标。在本档描述的产品中出现的其它商标，产品名称，和服务名称，均由其各自所有人拥有。

免责声明

您购买的产品、服务或特性应受您与珠海全志科技股份有限公司（“全志”）之间签署的商业合同和条款的约束。本档中描述的全部或部分产品、服务或特性可能不在您所购买或使用的范围内。使用前请认真阅读合同条款和相关说明，并严格遵循本档的使用说明。您将自行承担任何不当使用行为（包括但不限于如超压，超频，超温使用）造成的不利后果，全志概不负责。

本档作为使用指导仅供参考。由于产品版本升级或其他原因，本档内容有可能修改，如有变更，恕不另行通知。全志尽全力在本档中提供准确的信息，但并不确保内容完全没有错误，因使用本档而发生损害（包括但不限于间接的、偶然的、特殊的损失）或发生侵犯第三方权利事件，全志概不负责。本档中的所有陈述、信息和建议并不构成任何明示或暗示的保证或承诺。

本档未以明示或暗示或其他方式授予全志的任何专利或知识产权。在您实施方案或使用产品的过程中，可能需要获得第三方的权利许可。请您自行向第三方权利人获取相关的许可。全志不承担也不代为支付任何关于获取第三方许可的许可费或版税（专利税）。全志不对您所使用的第三方许可技术做出任何保证、赔偿或承担其他义务。