Melis RTOS 存储系统
开发指南

版本号: 1.0
发布日期: 2021.05.24

版本历史

版本号	日期	制/修订人	内容描述
1.0	2021.05.24	KPA0353	初始化添加 Melis RTOS 存储开发方法

目　录

1 概述

1.1 编写目的

此文档介绍从 Melis 4.0 开始，Melis RTOS 的存储开发方法。

1.2 使用范围

Allwinner 软件平台 Melis 4.0 以上版本。

1.3 相关人员

使用 Melis 4.0 以上系统的文件系统开发者、OTA 开发者

2 文件系统接口

Melis RTOS 支持 C 标准的文件读写接口 (如 fopen/fwrite/fclose)，以及 POSIX 标准的文件读写接口 (如 open/write/close)。

表 2-1: 文件接口表

接口	作用
fopen	打开文件句柄
fclose	关闭文件句柄
fread	读文件
fwrite	写文件
fseek	文件位置跳转
ftell	返回当前位置
open	打开文件句柄
close	关闭文件句柄
read	读文件
write	写文件
lseek	2GB 内文件位置跳转
lseek64	文件位置跳转
unlink	删除文件
remove	删除文件
rename	文件重命名
dup	复制文件描述符
fsync	同步文件
ftruncate	改变文件大小
ioctl	控制文件
mkdir	创建目录
rmdir	删除目录
opendir	打开目录
closedir	关闭目录
readdir	读取目录
stat	获取文件信息
statfs	获取文件系统信息
umount	卸载文件系统
mount	挂载文件系统

3 文件系统配置

3.1 术语介绍

表 3-1: 文件系统名词介绍表

接口	作用
DFS	rt-thread 精简版虚拟文件系统
littlefs	针对嵌入式设备的文件系统
ramfs	内存文件系统
devfs	rt-thread 设备文件系统
layerfs	DFS 转换层文件系统

当前 Melis RTOS 存在两套虚拟文件系统 (VFS)，一套模拟 Linux VFS，另外一套则源自 RT-Thread DFS。模拟 Linux 的 VFS 主要是为了支持 fat32, exfat, ntfs 等文件系统，适用于 TF 卡、U 盘等介质，不使用该类介质时，可以关闭，节省代码大小。采用 RT-Thread DFS 主要是为了使用 littlefs，ramfs，devfs，layerfs 等文件系统。

模拟 Linux 的 VFS 使用时，会通过 layerfs 与 DFS 进行适配。每当模拟 Linux 的 VFS 挂载一个文件系统系统，便会在 DFS 中挂载一个 layerfs 类型的文件系统。通过 layerfs，可以使用 DFS 的接口去使用 fat32, exfat, ntfs 等文件系统，以达到统一接口又兼容旧版本的目的。

3.2 DFS 配置

配置项如下：

```
make menuconfig
    Kernel Setup  --->
        Subsystem support  --->
            RT-Thread DFS Support  --->
                [*] rt-thread dfs virtual file system  #使能 RT-Thread DFS
```

3.2.1 littlefs 配置

littlefs 主要针对 flash 特性而开发的文件系统，在 Melis RTOS 系统中，主要用来挂载 nor flash 的用户分区，为用户提供 flash 文件读写功能，其配置项如下：

```
make menuconfig
    Kernel Setup  --->
        Subsystem support  --->
            RT-Thread DFS Support  --->
                [*] rt-thread dfs virtual file system  #使能 RT-Thread DFS
                [*]   Enable Little file system #  使能 littlefs
```

3.2.2 ramfs 配置

ramfs 的挂载介质为内存块，提供临时性的文件读写功能，配置项如下：

```
make menuconfig
    Kernel Setup  --->
        Subsystem support  --->
            RT-Thread DFS Support  --->
                [*] rt-thread dfs virtual file system  #使能 RT-Thread DFS
                [*]   Enable RAM file system
                (/tmp)  Ramfs mount path (NEW) #ramfs挂载路径
                (0x800000) Ramfs filesystem size (NEW)  #ramfs大小
                [*]     Ramfs support directory (NEW)  #ramfs支持目录
                [*]     Ramfs data slice (NEW)  ramfs支持数据分片
```

3.2.3 devfs 配置

devfs 是设备文件系统，主要用于管理设备，提供使用文件系统接口操作设备的功能，配置项如下：

```
make menuconfig
    Kernel Setup  --->
        Subsystem support  --->
            RT-Thread DFS Support  --->
                [*] rt-thread dfs virtual file system  #使能 RT-Thread DFS
                [*]   Using devfs for device objects   #使能 devfs
```

3.2.4 rootfs 配置

rootfs 为根目录挂载文件系统，其管理的数据为文件系统挂载表。通过 rootfs，可以获取当前挂载的所有文件系统，其配置项如下：

```
make menuconfig
    Kernel Setup  --->
        Subsystem support  --->
            RT-Thread DFS Support  --->
                [*] rt-thread dfs virtual file system  #使能 RT-Thread DFS
                [*]   Mount Melis rootfs file system for root directory
```

3.3 模拟 Linux 的 VFS

该 VFS 主要用于支持 U 盘、SD 卡等介质上的 fat,exfat, ntfs 等文件系统，其配置项如下:

```
make menuconfig
    Kernel Setup  --->
        File system support  --->
            [*] Support melis virtual file system  #使能VFS
            [*] Support devfs file system #使能对应的devfs
            [ ] Support minfs file system
            [ ] Support cdfs file system
            [ ] Support cdfs file system
            [ ] Support exFAT file system
            [ ] Support ntfs file system
            [ ] Support fat file system
```

除此之外，如需支持原 Melis 4.0 以下的设备管理框架，还需打开以下配置：

```
make menuconfig
    Kernel Setup  --->
        Melis Legacy Support  --->
            [*] Support Melis Legacy Driver Management #使能Melis 4.0以下系统的设备管理接口
            [*]   Support Melis Legacy DMS Device  # 添加devfs虚拟设备
```

如需通过 DFS 接口访问 ntfs，fatfs，exfat 等文件系统，需要使能 layerfs，其配置项如下：

```
make menuconfig
    Kernel Setup  --->
        Subsystem support  --->
            RT-Thread DFS Support  --->
                [*]   Mount Melis FS AS Dir in DFS (NEW) # 使能layerfs
                (/mnt/) Melis Layerfs Dir Path in DFS (NEW)  # 设置挂载路径
```

4 分区配置

分区配置是通过方案目录下的 sys_partition_${存储类型}.fex 文件进行配置。如需新增分区，可参考下列案例，添加一个表项：

```
[partition]
    name         = ROOTFS       //分区名
    size         = 8192         //分区大小，单位为512 byte
    downloadfile = "rootfs.fex"  //填充分区的文件，该文件需要在打包过程中放置到melis-v3.0\source\
    out\${board_name}\image目录下，同时必须以.fex为文件后缀
    user_type    = 0x8000
```

系统启动时，会获取系统分区信息，然后为每个分区在设备文件系统中创建一个设备节点，设备节点名格式为 “/dev/xxx”, xxx 表示在 sys_partition.fex 中配置的分区名称。通过该设备节点，开发人员可以直接读写分区。

4.1 注意事项

配置分区时，需要注意 rootfs 分区大小与 rootfs.ini 配置的大小要一致（注意单位）。且配置的所有分区大小加上 128kb(boot0) 不应超过 flash 总大小。

5 分区读写

对于裸分区读写，可直接通过设备节点的方式进行读写。

5.1 范例

```
int fd = open("/dev/UDISK", O_RDWR); //通过设备节点打开分区
write(fd, data, size);
close(fd);
```

5.2 注意事项

对于裸分区写，只能进行顺序写，无法进行随机写，也不可使用 lseek 进行跳转写入。

6 存储介质开发

6.1 NOR Flash

6.1.1 新增 NOR Flash 支持

NOR Flash 驱动核心代码位于 ekernel/drivers/hal/source/spinor/中，如需增加对新物料的支持，需要添加厂商对应的 C 文件，或者在对应文件中增加

```
struct nor_info
{
    char *model;
    unsigned char id[MAX_ID_LEN];
    unsigned int total_size;

    int flag;
#define SUPPORT_4K_ERASE_BLK        BIT(0)
#define SUPPORT_32K_ERASE_BLK       BIT(1)
#define SUPPORT_64K_ERASE_BLK       BIT(2)
#define SUPPORT_DUAL_READ           BIT(3)
#define SUPPORT_QUAD_READ           BIT(4)
#define SUPPORT_QUAD_WRITE          BIT(5)
#define SUPPORT_INDIVIDUAL_PROTECT  BIT(6)
#define SUPPORT_ALL_ERASE_BLK       (SUPPORT_4K_ERASE_BLK | \
                                     SUPPORT_32K_ERASE_BLK | \
                                     SUPPORT_64K_ERASE_BLK)
#define SUPPORT_GENERAL             (SUPPORT_ALL_ERASE_BLK | \
                                     SUPPORT_QUAD_WRITE | \
                                     SUPPORT_QUAD_READ | \
                                     SUPPORT_DUAL_READ)
#define USE_4K_ERASE                BIT(20)
#define USE_IO_PROG_X4              BIT(21)
#define USE_IO_READ_X2              BIT(22)
#define USE_IO_READ_X4              BIT(23)
};
```

指明新增物料信息，如物料名称，jedec id，容量，擦除块大小。

增加

```
struct nor_factory {
unsigned char factory;
unsigned int idt_cnt;
struct nor_info *idt;

int (*init)(struct nor_flash *nor);
void (*deinit)(struct nor_flash *nor);
```

```
    int (*init_lock)(struct nor_flash *nor);
    void (*deinit_lock)(struct nor_flash *nor);
    int (*lock)(struct nor_flash *nor, unsigned int addr, unsigned int len);
    int (*unlock)(struct nor_flash *nor, unsigned int addr, unsigned int len);
    bool (*islock)(struct nor_flash *nor, unsigned int addr, unsigned int len);
    int (*set_quad_mode)(struct nor_flash *nor);
    int (*set_4bytes_addr)(struct nor_flash *nor, int enable);

    struct nor_factory *next;
    };
```

并实现对应厂家芯片进入quad_mode，set_4bytes等状态的函数。

```
    static int nor_factory_register(void)
    {
        nor_register_factory_gd();
        nor_register_factory_mxic();
        nor_register_factory_winbond();
        nor_register_factory_xtx();
        nor_register_factory_esmt();
        nor_register_factory_fm();
        nor_register_factory_xmc();
        nor_register_factory_puya();
        nor_register_factory_zetta();
        return 0;
    }
```

并确保芯片厂家对应的信息在nor_factory_register中注册

6.1.2 NOR Flash 锁保护开发

6.1.2.1 独立块锁保护

宏CONFIG_DRIVERS_SPINOR_WRITE_LOCK 用于配置 NOR Flash 独立块锁保护，如该款物料支持独立块锁保护，请按下列步骤进行开发。1. 在芯片对应的 struct nor_factory 结构体，实现对应的 lock 和 unlock 功能函数即可。

6.1.2.2 注意事项

对于新款 Nor Flash 物料的锁保护开发，除了需要在 Melis RTOS 系统的 Flash 驱动进行适配，还需要在 bootloader 中进行适配，否则将无法正常烧录和启动。在 bootloader 中的适配，请联系全志工程师予以协助。

6.2 SDMMC Card

SDMMC Card 使能配置如下:

```
# 选上SDMMC Card驱动
make menuconfig
    Kernel Setup  --->
        Drivers Setup  --->
            Melis Source Support  --->
                [*] Support SDMMC Driver # 使能SDMMC Card驱动

#  SDMMC Card设备注册方式较为特殊，需打开以下配置
make menuconfig
    Kernel Setup  --->
        Drivers Setup  --->
            Melis Legacy Support  --->
                [*] Support Melis Legacy Driver Management #使能Melis 4.0以下系统的设备管理接口
                [*]   Support Melis Legacy DMS Device  # 添加devfs虚拟设备

# 选上VFS支持，如需挂载fat或者ntfs等文件系统，请自行选上
make menuconfig
    Kernel Setup  --->
        File system support  --->
            [*] Support melis virtual file system  #使能VFS
            [*] Support devfs file system #使能对应的devfs
```

除此之外，提供了 SDMMC Card 的缓存功能，其实现了预读、读写相邻块合并、数据缓存等功能，配置项如下:

```
make menuconfig
    Kernel Setup  --->
        Drivers Setup  --->
            [*] support sdmmc cache  # 使能SDMMC数据缓存
            (1024) sdmmc cache size (KB) (NEW)  #设置数据缓存大小
            (14)  sdmmc cache writeback thread priority (NEW)#缓存数据回刷任务的优先级
            [ ]   support sdmmc cache information command (NEW) # 数据缓存调试命令
```

6.3 U 盘存储

F133 分 usb0 和 usb1；其中 usb0 支持 OTG，usb1 只支持 host 模式

```
系统配置:
make menuconfig
    Kernel Setup --->
        Drivers Setup --->
            Melis Source Support
                [*] OS USB Dirver  --->
                    DRIVER_USB_MANAGER  ---> # 打开usb manager功能。
                    [*] Support usb host ehci1 # 使能usb1
                    [*] Support usb host core # 使能usb0
            SoC HAL Drivers  --->
                [*] USB Drivers  --->
```

```
                    USB HOST  --->
                        [*] enable usb host driver # 使能usb host控制器
                        [*]   enable HCI hal APIs test command
                         [*]   USB Mass Storage support # U盘驱动
                    [*]   USB Manager support # Usb manager驱动代码。
```

```
配置文件：
sys_config_${存储类型}.fex
    [usbc0]：控制器0的配置。
    usb_used：USB使能标志。置1，表示系统中USB模块可用,置0,则表示系统USB禁用。
    usb_port_type：USB端口的使用情况。 0：device only;1：host only;2：OTG
    usb_detect_type：USB端口的检查方式。0：不做检测;1：vbus/id检查;2：id/dpdm检查
    usb_detect_mode：USB端口的检查方式。0：线程轮询;1：id中断触发
    usb_id_gpio：USB ID pin脚配置。具体请参考gpio配置说明。
    usb_det_vbus_gpio：USB DET_VBUS pin脚配置。具体请参考gpio配置说明。
    usb_drv_vbus_gpio：USB DRY_VBUS pin脚配置。具体请参考gpio配置说明。
    usb_drv_vbus_type:  vbus设置方式。0：无；1：gpio；2：axp。
    usb_det_vbus_gpio: "axp_ctrl",表示axp 提供
    usb_restrict_gpio  usb限流控制pin
    usb_restric_flag:  usb限流标置
    usbh_driver_level:  usb做host时驱动能力0x0-0xf
    usbd_driver_level:  usb做device时驱动能力0x0-0xf
    usbh_irq_flag:  usb host中断标志 设置为0x4000可为唤醒源，设置为0不具有唤醒功能。
    usbd_irq_flag: usb device中断标志 设置为0x4000可为唤醒源，设置为0不具有唤醒功能。
```

7 测试

7.1 rwcheck 校验测试

rwcheck 命令用于测试文件系统、存储驱动读写数据的正确性。配置方法：

```
make menuconfig
    Kernel Setup  --->
        Subsystem support  --->
            Command shell  --->
                Commands  --->
                    Iobox Command  --->
                        [*] Rwcheck command
```

其使用方法如下：

```
rwcheck <-d dir> [-h] [-t times] [-s size] [-p percent] [-m jobs]
-d : 测试目录
-h : 帮助信息
-t : 测试次数
-s : 测试文件大小
-p : 测试数据占文件系统百分比
-m : 测试任务个数
```

7.2 rwspeed 性能测试

rwspeed 命令用于测试文件系统、存储驱动读写数据的性能。配置方法如下：

```
make menuconfig
    Kernel Setup  --->
        Subsystem support  --->
            Command shell  --->
                Commands  --->
                    Iobox Command  --->
                        [*] Rwspeed command
```

其使用方法如下：

```
rwspeed <-d dir> [-h] [-t avg] [-s size]
-d : 测试目录
-h : 帮助信息
-t : 测试次数
-s : 测试文件大小
-b : 每笔读写数据大小
```

```
-m ： 测试任务个数
```

著作权声明

商标声明

免责声明