**exFat开源代码搭载总结**

**历史记录**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本号 | 日期 | 作者/修订者 | 修改内容 | 评审者 |
| 1.0.0 | 2020/11/18 | xxxxxx | exFat初版 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

**目录**

[1. 概要 4](#_Toc56603845)

[1.1. 目的与背景 4](#_Toc56603846)

[1.2. 参考文献 4](#_Toc56603847)

[1.3. 术语和定义 5](#_Toc56603848)

[2. 架构设计 5](#_Toc56603849)

[2.1. 功能点说明 5](#_Toc56603850)

[2.2. 软件架构 5](#_Toc56603851)

[2.3. 硬件架构 5](#_Toc56603852)

[2.4. 系统裁剪 5](#_Toc56603853)

[3. 详细设计 5](#_Toc56603854)

[3.1. 软件模块划分 5](#_Toc56603855)

[3.2. 模块间关系设计 5](#_Toc56603856)

[3.3. 重要功能函数设计 6](#_Toc56603857)

[3.3.1 exFat检查 6](#_Toc56603858)

[3.3.2 exFat挂载 6](#_Toc56603859)

[3.3.3 exFat格式化 7](#_Toc56603860)

[3.4. 出错处理及安全保密设计 8](#_Toc56603861)

[3.5. 维护设计 8](#_Toc56603862)

# 概要

## 目的与背景

目的：exFat在Framework层功能的实现。

背景：Android10不支持exFat。

## 参考文献

|  |  |
| --- | --- |
| No | 文献名或者地址 |
| 1 | https://github.com/relan/exfat |
|  |  |
|  |  |
|  |  |
|  |  |

## 术语和定义

|  |  |
| --- | --- |
| 缩写词或者术语 | 定义 |
| Vold | Volume Daemon |
| exFat | Extended File Allocation Table |
| fsck | File system check |
| mkfs | Make file system |
| e2fsprogs | Ext2 Filesystems Utilities |

# 架构设计

## 功能点说明

|  |  |  |
| --- | --- | --- |
| 功能 | 描述 | 备注 |
| exFat实现 | 实现exFat检查、挂载、格式化操作 | 实现文件Exfat.cpp，Exfat.h |

## 软件架构

## 硬件架构

## 系统裁剪

# 详细设计

## 软件模块划分

Android原有的Vold模块中添加exFat支持。

## 模块间关系设计

模块间的依赖关系跟Android原有的Vold模块相同。

## 重要功能函数设计

首先需要通过下面方式获取exFat的开源代码:

git clone https://github.com/relan/exfat.git

### 3.3.1 exFat检查

Vold模块在Sdcard挂载前需要先检查exFat是否可以正常挂载。具体检查代码如下:

|  |  |
| --- | --- |
| 文件名 | 代码 |
| Exfat.cpp | static const char\* kFsckPath = "/system/bin/fsck.exfat";  status\_t Check(const std::string& source) {  std::vector<std::string> cmd;  cmd.push\_back(kFsckPath);  cmd.push\_back(source);  int rc = ForkExecvp(cmd, nullptr, sFsckUntrustedContext);  if (rc == 0) {  LOG(INFO) << "Check OK";  return 0;  } else {  LOG(ERROR) << "Check failed (code " << rc << ")";  errno = EIO;  return -1;  }  } |

可以看出，exFat检查是通过fsck.exfat命令来实现的，该命令的具体实现代码请阅读参考文献[1]。

### 3.3.2 exFat挂载

当exFat检查OK之后，需要调用系统内核的mount来实现挂载功能。具体代码如下:

|  |  |
| --- | --- |
| 文件名 | 代码 |
| Exfat.cpp | #include <sys/mount.h>  ……  status\_t Mount(const std::string& source, const std::string& target, int ownerUid, int ownerGid,  int permMask) {  int mountFlags = MS\_NODEV | MS\_NOSUID | MS\_DIRSYNC | MS\_NOATIME | MS\_NOEXEC;  auto mountData = android::base::StringPrintf("uid=%d,gid=%d,fmask=%o,dmask=%o", ownerUid,  ownerGid, permMask, permMask);  LOG(INFO) << "Mount source="<<source.c\_str()<<",target="<<target.c\_str();  if (mount(source.c\_str(), target.c\_str(), "exfat", mountFlags, mountData.c\_str()) == 0) {  return 0;  }  PLOG(ERROR) << "Mount failed; attempting read-only";  mountFlags |= MS\_RDONLY;  if (mount(source.c\_str(), target.c\_str(), "exfat", mountFlags, mountData.c\_str()) == 0) {  return 0;  }  return -1;  } |

### 3.3.3 exFat格式化

exFat格式化的具体实现代码如下:

|  |  |
| --- | --- |
| 文件名 | 代码 |
| Exfat.cpp | static const char\* kMkfsPath = "/system/bin/mkfs.exfat";  ……  status\_t Format(const std::string& source) {  std::vector<std::string> cmd;  cmd.push\_back(kMkfsPath);  cmd.push\_back("-n");  cmd.push\_back("android");  cmd.push\_back(source);  int rc = ForkExecvp(cmd);  if (rc == 0) {  LOG(INFO) << "Format OK";  return 0;  } else {  LOG(ERROR) << "Format failed (code " << rc << ")";  errno = EIO;  return -1;  }  return 0;  } |

可以看出，exFat格式化是通过mkfs.exfat命令来实现的，该命令的具体实现代码请阅读参考文献[1]。

## 出错处理及安全保密设计

暂无设计

## 维护设计

暂无设计