|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **昆山通祐电梯有限公司** | | **文档编号** | **版本号** | **密级** |
|  |  |  |
| **文档名称** |  | | **日期** |  |

**嵌入式视觉传感器嵌入式端内核和文件系统双备份说明书**

**文档作者： 马洁 日期：2017/7/21**

**审 核： 日期：**

**批 准： 日期：**

**机器视觉及人工智能联合实验室 版权所有**

**内部资料 注意保密**

文档历史发放及记录

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **序号** | **变更（+/-）说明** | **作者** | **版本号** | **日期** | **批准** |
| 1 | 创建 | 马洁 | V1.0 | 2017.7.21 |  |
| 2 |  |  |  |  |  |
| 3 |  |  |  |  |  |
| 4 |  |  |  |  |  |
| 5 |  |  |  |  |  |
| 6 |  |  |  |  |  |
| 7 |  |  |  |  |  |

[共享缓存区设计说明书 4](#_Toc488440496)

[引言 4](#_Toc488440497)

[1. 编写目的 4](#_Toc488440498)

[2. 项目背景 4](#_Toc488440499)

[3. 定义 4](#_Toc488440500)

[4. 参考资料 5](#_Toc488440501)

[总体设计 6](#_Toc488440502)

[1. 需求概述 6](#_Toc488440503)

[2. 软件结构 6](#_Toc488440504)

[共享缓存区设计描述 7](#_Toc488440505)

[1. 目的 7](#_Toc488440506)

[2. 分区 8](#_Toc488440507)

[3. 缓存块数据结构 9](#_Toc488440508)

[4. 缓存区流程 10](#_Toc488440509)

[5. 相关结构体 14](#_Toc488440510)

[6. 存储分配 14](#_Toc488440511)

[7. 测试要点 14](#_Toc488440512)

内核和文件系统双备份设计说明书

# 引言

## 编写目的

【阐明编写需求说明书的目的，指明读者对象】

本概要说明书仅供机器视觉及人工智能联合实验室部门内部使用，任何人未经批准，不得外发及泄露。

本设计说明书对**嵌入式视觉传感器的嵌入式端共享缓存区**进行设计说明而编写，理清整体设计思路，为后续客户端软件开发提供可靠、全面的信息支持以及产品开发完成后的验证。

## 项目背景

【包括：

1. 项目委托单位、开发单位、主管部门;
2. 该软件系统与其他系统的关系。

】

项目委托单位：西安电子科技大学

开发单位：西安电子科技大学软件学院嵌入式技术与智能视觉处理研究中心

主管部门：通祐电梯

该软件系统是针对嵌入式视觉传感器的管理中心、参数中心、检测模块、智能处理模块、采集模块、升级模块而开发的嵌入式软件系统。

## 定义

【包括：列出文档中所用到的专业术语的定义和缩写词的原文。】

## 参考资料

【包括：列出有关资料的作者、标题、编号、发表日期、出版单位或资料来源。

* + 1. 项目经核准的计划任务书、合同或上级机关的批文;
    2. 项目开发计划；
    3. 需求规格说明书
    4. 测试计划（初稿）
    5. 用户操作手册（初稿）
    6. 文档所引用的资料、标准、和规范。

】

嵌入式视觉传感器需求说明书-V1.3——通祐电梯，人工智能联合实验室

传感器嵌入式软件概要设计书\_V1.4——通祐电梯，人工智能联合实验室

# 内核和文件系统双备份设计描述

## 目的

内核和文件系统双备份是为了防止升级或者某些操作导致内核启动不起来或者文件系统挂载不上造成的嵌入式设备打不开的问题。当一份内核文件系统崩溃之后，双备份机制就会启动另外一份内核和文件系统。

## mtd分区

将flash分为6个mtd分区：uboot，version(1M)，kernel\_0，fs\_0，kernel\_1，fs\_1。Version分区中分出4个flash块来保存两个内核标志位和两个内核对应的启动flash块。

MTD分区图

1M

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Uboot | Version | Kernel\_0 | Fs\_0 | Kernel\_1 | Fs\_1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Kernel\_0内核标志位(0或1) | 计算kernel\_0启动次数的一块flash | Kernel\_1内核标志位(0或1) | 计算kernel\_1启动次数的一块flash | 其他 |

64K 64K 64K 64K 768K

* **说明：**

Nor flash写某个地址之前需要进行擦除，而擦除的单位是块。当前的nor flash每一块大小是64K。所以在version分区中拿出来四个flash块来保存内核标志位和启动次数。

* **内核标志位：**

使用内核标志位来标记哪一个内核是最新的。用1表示是最新的，用0表示不是最新的。正常情况下，uboot需要启动最新的内核，也就是内核标志被置1的对应的内核。只有两种情况会修改内核标志位：一种是最新的内核或文件系统连续5次启动不起来，就启动另一份内核和文件系统。同时，将两个内核标志位的值互换。另一种是升级内核和文件系统，在升级时将两个内核标志位的值互换。

* **计算内核启动次数的块：**

在Uboot决定启动某个内核时，

1. 首先判断该内核是否已经启动了5次而未成功。判断的条件是在其对应的flash块中查看已经被写过的最后5位是否全部为0x11。
2. 如果是，则交换两个内核标志位，断电重启。如果不是，则在其对应的flash块中写一个0x11。
3. 如果该内核和文件系统启动成功，则在启动完成之时在该块中写一个0x22。如果启动没有成功，则断电重启。

## 双备份流程

双备份流程图

* **说明：**

内核和文件系统是绑定的，若是一组内核和文件系统连续5次启动不起来，那就启动另一组内核和文件系统。在自启动程序中，找到最新的内核对应的flash，在没有写过的第一位写0x22。要是uboot和应用程序中发现改块已经写满了，则擦除整个flash块。

* **uboot：**

1. 找出最新的内核。
2. 将该内核对应的flash块中，在没有被写过的第一位写0x11。如果改块已经被写满了，则擦除整个flash块，然后在第一位写0x11.
3. 判断是否被写过的最后5位为0x11，若是，则将两个内核标志位互换，若不是，则将最新的内核读取内存中，将ubootargs中的根文件系统挂载到最新的文件系统对应的mtd中，然后启动该内核。

* **内核和文件系统启动成功：**

在内核和文件系统启动成功之后，会自动运行一个程序。该程序的功能为:

1. 找出最新的内核。
2. 将该内核对应的flash块中，在没有被写过的第一位写0x22。如果该块已经被写满了，则擦除整个flash块，然后在第一位写0x22。

* **升级进程：**

1. 找出非最新的内核。
2. 将该内核对应的flash块中，在没有被写过的第一位写0x22。如果改块已经被写满了，则擦除整个flash块，然后在第一位写0x22.
3. 交换两个内核标志位。
4. 将从客户端接收到的内核和文件系统镜像覆盖找出的内核和文件系统对应的flash中。

## 测试要点

* **uboot**：

测试在是否可以正常读取出两个内核对应的内核标志位。

是否可以正常在内核对应的flash块中写入0x11。

是否可以正常擦出flash块。

是否可以正常交换两个内核标志位。

是否可以正常修改bootargs。

* **嵌入式正常启动**：

测试是否可以在自启动程序中找到启动的是哪个内核。

是否可以正常在内核对应的flash块中写入0x22.