User Manual

M22A-F/N/U20 系列 ARM 嵌入式工业控制模块

文件管理系统及相关存储设备函数使用手册

UM050111 V1.80 Date: 2008/03/08

产品用户手册

类别	内容
关键词	MiniARM® M22A 系列文件系统 API
摘　要	MiniARM® M22A 系列文件系统使用说明。

广州致远电子有限公司

ZHIYUAN ELECTRONICS

修订历史

版本	日期	原因
V1.00	2006-08-24	创建文档
V1.10	2006-08-30	修改域名和邮件地址
V1.20	2007-09-17	修改版面
V1.30	2007-09-21	第 14 页，修改表 4.21、4.22 对函数的错误调用
V1.40	2007-10-09	第 13 页，表 4.20，将 OSAllCacheWriteBack 函数返回类型修正为 void
V1.50	2007-11-24	第 14 页，将函数 format 修正为 OSFormat
V1.60	2007-12-17	修改销售服务网络联系方式
V1.70	2008-01-15	文档纠错
V1.80	2008-03-08	版面修改

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

销售与服务网络（一）

广州周立功单片机发展有限公司

地址：广州市天河北路 689 号光大银行大厦 12 楼 F4 邮编：510630
电话：(020)38730916 38730917 38730972 38730976 38730977
传真：(020)38730925
网址：www.zlgmcu.com

广州专卖店

地址：广州市天河区新赛格电子城 203-204 室
电话：(020)87578634 87569917
传真：(020)87578842

南京周立功

地址：南京市珠江路 280 号珠江大厦 2006 室
电话：(025)83613221 83613271 83603500
传真：(025)83613271

北京周立功

地址：北京市海淀区知春路 113 号银网中心 A 座 1207-1208 室（中发电子市场斜对面）
电话：(010)62536178 62536179 82628073
传真：(010)82614433

重庆周立功

地址：重庆市石桥铺科园一路二号大西洋国际大厦（赛格电子市场）1611 室
电话：(023)68796438 68796439
传真：(023)68796439

杭州周立功

地址：杭州市登云路 428 号浙江时代电子市场 205 号
电话：(0571)88009205 88009932 88009933
传真：(0571)88009204

成都周立功

地址：成都市一环路南二段 1 号数码同人港 401 室（磨子桥立交西北角）
电话：(028)85439836 85437446
传真：(028)85437896

深圳周立功

地址：深圳市深南中路 2070 号电子科技大厦 A 座 24 楼 2403 室
电话：(0755)83781788（5 线）
传真：(0755)83793285

武汉周立功

地址：武汉市洪山区广埠屯珞瑜路 158 号 12128 室（华中电脑数码市场）
电话：(027)87168497 87168297 87168397
传真：(027)87163755

上海周立功

地址：上海市北京东路 668 号科技京城东座 7E 室
电话：(021)53083452 53083453 53083496
传真：(021)53083491

西安办事处

地址：西安市长安北路 54 号太平洋大厦 1201 室
电话：(029)87881296 83063000 87881295
传真：(029)87880865

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

销售与服务网络（二）

广州致远电子有限公司

地址：广州市天河区车陂路黄洲工业区 3 栋 2 楼
邮编：510660
传真：(020)38601859
网址：www.embedtools.com （嵌入式系统事业部）
www.embedcontrol.com （工控网络事业部）
www.ecardsys.com （楼宇自动化事业部）

技术支持：

CAN-bus：

电话：(020)22644381 22644382 22644253
邮箱：can.support@embedcontrol.com

iCAN 及模块：

电话：(020)28872344 22644373
邮箱：ican@embedcontrol.com

MiniARM：

电话：(020)28872684
邮箱：miniarm.support@embedtools.com

以太网及无线：

电话：(02 0)22644385 22644386
邮箱：wireless@embedcontrol.com
ethernet.support@embedcontrol.com

编程器：

电话：(020)38681856 28872449
邮箱：programmer@embedtools.com

分析仪器：

电话：(020)22644375 28872624 28872345
邮箱：tools@embedtools.com

ARM 嵌入式系统：

电话：(020)28872347 28872377 22644383 22644384
邮箱：arm.support@zlgmcu.com

楼宇自动化：

电话：(020)22644376 22644389
邮箱：mjs.support@ecardsys.com
mifare.support@zlgmcu.com

销售：

电话：(020)22644249 22644399 28872524 28872342
28872349 28872569 28872573

维修：

电话：(020)22644245

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

目　录

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

1. 适用范围

此函数参考手册适用于 MiniARM® M22A 系列嵌入式工控模块。

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

2. 描述

MiniARM® M22A 系列嵌入式工控模块固化了我公司自主开发的文件管理系统——ZLG/FS，它与 FAT12、FAT16 和 FAT32 文件系统高度兼容，同时，兼容多种介质，如 CF 卡、U 盘等，ZLG/FS 提供一个底层驱动程序的接口，用户只需要提供相应介质的驱动就可以在该介质上使用 ZLG/FS。

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

3. 特征

利用 MiniARM® M22A 系列嵌入式工控模块提供的文件系统，用户可以完成有关文件系统的大部分操作。该文件系统的特点如下：

- 文件系统已固化到产品中；
- 兼容 FAT12、FAT16 和 FAT32；
- 支持多种介质，而且支持多种介质同时使用；
- 支持多个逻辑盘；
- 支持文件读/写和目录操作；
- 支持树型目录结构，子目录层数不受限制；
- 支持文件属性中的时间属性；
- 支持长文件名；

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

4. 文件系统 API 函数说明

表 4.1 文件系统 API 函数一览表

函数名称	功能描述
OSFileOpen	以指定方式打开文件
OSFileClose	关闭指定文件
OSFileRead	读取文件
OSFileWrite	写文件
OSFileCloseAll	关闭所有打开的文件
OSFileEof	判断文件是否读\写到文件尾
OSFileSeek	移动文件读\写位置
OSRemoveFile	删除文件
OSRenameFile	文件改名
OSChangeDrive	改变当前逻辑盘
OSChangeDir	改变当前目录
OSMakeDir	建立目录
OSRemoveDir	删除目录
Osmount	加载卷（已分配盘符），允许读写
OSumount	卸载卷（已分配盘符），禁止读写
OSAddFileDriver	增加一个底层驱动程序
OSRemoveFileDriver	删除一个底层驱动程序
OSAllCacheWriteBack	把所有已改变的扇区写回逻辑盘
OSFormat	格式化逻辑卷
Format	格式化逻辑卷
OSFindFirst	在指定目录查看第一个文件/目录名称
OSFindNext	在指定目录查看下一个文件/目录名称
OSChangeCodePage	改变本地编码
OSStrupr	把字符串转换成大写
File_init	文件系统初始化
OSGetFileSize	获得指定文件大小
OSGetFileDataTime	获得指定文件最后写时间
OSGetFileOffset	获得文件当前读写位置

在文件系统中定义了多个返回字节，在使用过程中，用户可以通过这些标志来判断程序的执行情况，如程序清单 4.1 所示。

程序清单 4.1 文件系统返回值

```
/* 函数返回值 */
#define RETURN_OK                0x00    /* 操作成功          */
#define NOT_FIND_DISK            0x01    /* 逻辑盘不存在      */
#define DISK_FULL                0x02    /* 逻辑盘满          */
#define SECTOR_NOT_IN_CACHE      0x03    /* 扇区没有被 cache  */
```


©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

```
#define NOT_EMPTY_CACHE          0x04    /*  没有空闲 cache          */
#define SECTOR_READ_ERR          0x05    /*  读扇区错误              */
#define CLUSTER_NOT_IN_DISK      0x06    /*  逻辑盘中没有此簇        */
#define NOT_FIND_FDT             0x07    /*  没有发现文件(目录)      */
#define NOT_FAT_DISK             0x08    /*  非 FAT 文件系统         */
#define FDT_OVER                 0x09    /* FDT 索引超出范围         */
#define FDT_EXISTS               0x0a    /*  文件(目录)已经存在      */
#define ROOT_FDT_FULL            0x0b    /*  根目录满                */
#define DIR_EMPTY                0x0C    /*  目录空                  */
#define DIR_NOT_EMPTY            0x0d    /*  目录不空                */
#define PATH_NOT_FIND            0x0e    /*  路径未找到              */
#define FAT_ERR                  0x0f    /* FAT 表错误               */
#define FILE_NAME_ERR            0x10    /*  文件(目录)名错误        */
#define FILE_EOF                 0x11    /*  文件结束                */
#define FILE_LOCK                0x12    /*  文件被锁定              */
#define NOT_FIND_FILE            0x13    /*  没有发现指定文件        */
#define NOT_FIND_DIR             0x14    /*  没有发现指定目录        */
#define GET_TIME_ERR             0x15    /*  获取时间错误            */
#define DISK_NO_FORMAT           0x16    /*  逻辑盘没有格式化        */
#define NO_MEMERY                0x17    /*  内存不足                */
#define FIND_FILE                0x18    /*  找到文件                */
#define FIND_DIR                 0x19    /*  找到目录                */
#define FDT_EOF                  0x20    /* FTD 表结束               */

#define NOT_RUN                  0xfd    /*  命令未执行              */
#define BAD_COMMAND              0xfe    /*  错误命令                */
#define PARAMETER_ERR            0xff    /*  非法参数                */
```

表 4.2　File_init 函数

函数原型	**void　File_init(void)**
函数功能	文件系统初始化
入口参数	无
出口参数	无
调用示例	File_init();　　// 初始化文件系统

表 4.3　OSFileOpen 函数

函数原型	**HANDLE　OSFileOpen(char　*DirFileName, char　*Type)**
函数功能	以指定方式打开文件
入口参数	DirFileName　文件名 Type　打开方式：　“R” ：只读方式 “RW”：读写方式
出口参数	文件句柄 Not_Open_FILE　不能打开该文件
说明	如果目录下没有该文件，则系统会在该目录下创建这个文件。

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

调用示例	HANDLE fp;
	// 打开文件 Test.txt
	fp = OSFileOpen("A:\\Test.txt","RW");

表 4.4 OSFileClose 函数

函数原型	**uint8 OSFileClose(HANDLE Handle)**
函数功能	关闭指定文件
入口参数	Handle 文件句柄
出口参数	RETURN_OK 成功 其它参考关于文件系统返回值的说明

调用示例	HANDLE fp;
	// 打开文件 Test.txt
	fp = OSFileOpen("A:\\Test.txt","RW");
	OSFileClose(fp);

表 4.5 OSFileRead 函数

函数原型	**uint32 OSFileRead(void *Buf, uint32 Size, HANDLE Handle)**
函数功能	读取文件
入口参数	Buf 数据接收缓冲区首地址 Size 读取字节数 Handle 文件句柄
出口参数	实际读到的字节数

调用示例	// 从文件句柄 fp 所对应的文件中读取 50 字节数据，接收缓冲区为 Data
	LenFile = OSFileRead((uint8 *)Data, 50, fp);

表 4.6 OSFileWrite 函数

函数原型	**uint32 OSFileWrite(void *Buf, uint32 Size, HANDLE Handle)**
函数功能	向文件中写入数据
入口参数	Buf 发送数据缓冲区首地址 Size 写入字节数 Handle 文件句柄
出口参数	实际写入的字节数

调用示例	// 向文件句柄 fp 所对应的文件中写入字符串“Zlgmcu”
	sprintf(WriteBuf,"Zlgmcu");
	OSFileWrite(WriteBuf,sizeof(WriteBuf),fp);

表 4.7 OSFileCloseAll 函数

函数原型	**uint8 OSFileCloseAll(void)**
函数功能	关闭所有打开的文件
入口参数	无
出口参数	NOT_RUN 未执行 RETURN_OK 成功
调用示例	OSFileCloseAll( ); // 关闭所有打开文件

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

表 4.8 OSFileEof 函数

函数原型	**uint8 OSFileEof(HANDLE Handle)**
函数功能	判断文件是否到读\写到文件尾
入口参数	Handle 文件句柄
出口参数	0 否 1 是

调用示例	// 根据 state 判断：是否已经读\写到文件句柄 fp 所对应的文件尾
	state = OSFileEof(fp);

表 4.9 OSFileSeek 函数

函数原型	**uint8 OSFileSeek(HANDLE Handle, int32 offset, uint8 Whence)**
函数功能	移动文件读\写位置
入口参数	Handle 文件句柄 offset 移动偏移量 Whence 移动模式 SEEK_SET 从文件头计算 SEEK_CUR 从当前位置计算 SEEK_END 从文件尾计算
出口参数	参考关于文件系统返回值的说明

调用示例	// 将文件指针指向 fp1 所对应文件的末尾，然后可以追加写入数据
	OSFileSeek(fp1 , 0, SEEK_END);

表 4.10 OSRemoveFile 函数

函数原型	**uint8 OSRemoveFile(char *DirFileName)**
函数功能	删除文件
入口参数	DirFileName 文件名
出口参数	RETURN_OK 成功 其它参考关于文件系统返回值的说明

调用示例	// 删除根目录下的文件 abc.txt
	OSRemoveFile(“abc.txt”);

表 4.11 OSRenameFile 函数

函数原型	**uint8 OSRenameFile (char *OldName, char *NewName)**
函数功能	更改文件名
入口参数	OldName 旧文件名 NewName 新文件名
出口参数	RETURN_OK 成功 其它参考关于文件系统返回值的说明
调用示例	// 将文件 STORE.TXT 更名为 SEND.TXT OSRenameFile ("STORE.TXT", "SEND.TXT");

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

表 4.12 OSGetFileSize 函数

函数原型	**uint32 OSGetFileSize(char *DirFileName)**
函数功能	获得指定文件大小
入口参数	DirFileName 用户使用的文件名
出口参数	文件大小。若文件不存在返回 0

调用示例	uint32 FileSize;
	// 获取文件 Test.txt 的大小
	FileSize = OSGetFileSize("A:\\Test.txt");

表 4.13 OSGetFileDataTime 函数

函数原型	**uint8 OSGetFileDataTime(DATE_TIME *Data, char *DirFileName)**
函数功能	获得指定文件最后修改时间
入口参数	Data 文件修改时间 DirFileName 用户使用的文件名
出口参数	RETURN_OK 成功 其它参考关于文件系统返回值的说明
说明	/*时间格式 */ typedef struct _DATE_TIME { uint16 da_year; /* 公元年 */ uint8 da_mon; /* 月 */ uint8 da_day; /* 月中日期 */ uint8 da_dow; /* 星期中日期 */ uint8 ti_hour; /* 时 */ uint8 ti_min; /* 分 */ uint8 ti_sec; /* 秒 */ uint8 ti_hund; /* 百分之一秒 */ }DATE_TIME;

调用示例	DATE_TIME FileTime;
	// 获取文件 Test.txt 的最后修改时间
	OSGetFileDataTime(&FileTime,"A:\\Test.txt");

表 4.14 OSGetFileOffset 函数

函数原型	**uint32 OSGetFileOffset(HANDLE Handle, uint8 *err)**
函数功能	获得文件当前读写位置位置
入口参数	Handle 文件句柄 err 用于返回出错信息
出口参数	文件读、写位置(偏移量)

调用示例	uint32 offset;
	// 获得句柄 fp 所对应文件的当前读写位置
	offset = OSGetFileOffset(fp,&err);

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

表 4.15 OSChangeDrive 函数

函数原型	**uint8 OSChangeDrive(char *Drive)**
函数功能	改变当前逻辑盘符
入口参数	Drive 逻辑盘符字符串
出口参数	RETURN_OK 成功 NOT_FIND_DISK 逻辑盘不存在 PARAMETER_ERR 非法参数

调用示例	// 将当前逻辑盘符切换到 D 盘
	state = OSChangeDrive(“D:\\”);

表 4.16 OSChangeDir 函数

函数原型	**uint8 OSChangeDir(char *Path)**
函数功能	改变当前路径
入口参数	Path 路径名
出口参数	RETURN_OK 成功 其它参考关于文件系统返回值的说明
说明	系统默认的当前路径是根目录，可以通过该函数改变系统当前路径。

调用示例	/* 文件 test.txt 的路径：A:\Test1\Test2\test.txt */
	// 使用默认路径访问 test.txt 文件的方法
	fp = OSFileOpen("A:\\Test1\\Test2\\test.txt","RW"); // 访问 test.txt 文件
	
	// 更改当前路径后，访问 test.txt 文件的方法
	OSChangeDir("A:\\Test1\\Test2"); // 更改当前路径
	fp = OSFileOpen("test.txt","RW"); // 访问 test.txt 文件

表 4.17 OSMakeDir 函数

函数原型	**uint8 OSMakeDir(char *Path)**
函数功能	建立目录
入口参数	Path 绝对路径名
出口参数	RETURN_OK 成功 其它参考关于文件系统返回值的说明

调用示例	// 在根目录下建立一个目录 Dir
	OSMakeDir("A:\\Dir");

表 4.18 OSRemoveDir 函数

函数原型	**uint8 OSRemoveDir(char *Path)**
函数功能	删除目录
入口参数	Path 绝对路径名
出口参数	RETURN_OK 成功 其它参考关于文件系统返回值的说明
调用示例	// 将根目下的 Dir 文件夹删除 OSRemoveDir ("A:\\Dir");

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

表 4.19 OSAddFileDriver 函数

函数原型	**void OSAddFileDriver(void * DiskCommand, void *RsvdForLow)**
函数功能	增加一个底层驱动程序
入口参数	DiskCommand 驱动程序接口函数 RsvdForLow 底层驱动的参数
出口参数	无
说明	操作 CF 卡/IDE 时，*RsvdForLow ＝ 0：操作模式为主模式 *RsvdForLow ＝ 1：操作模式为从模式

调用示例	void *DiskCommand;
	uint8 CFSel = 0;
	// 获取 CF 卡驱动程序地址
	DiskCommand = GetCFCommand();
	// 加载 CF 卡驱动,使用主模式操作
	OSAddFileDriver(DiskCommand, &CFSel);

表 4.20 OSRemoveFileDriver 函数

函数原型	**void OSRemoveFileDriver(void *DiskCommand, void *RsvdForLow)**
函数功能	删除一个底层驱动程序
入口参数	DiskCommand 驱动程序接口函数 RsvdForLow 底层驱动的参数
出口参数	无
说明	操作 CF 卡/IDE 时，*RsvdForLow ＝ 0：操作模式为主模式 *RsvdForLow ＝ 1：操作模式为从模式

调用示例	void *DiskCommand;
	uint8 CFSel = 0;
	DiskCommand = GetCFCommand(); // 获取CF 卡驱动程序地址
	// 加载CF 卡驱动,使用主模式操作
	OSAddFileDriver(DiskCommand, &CFSel);
	……
	OSRemoveFileDriver (DiskCommand, &CFSel); // 删除 CF 卡驱动

表 4.21 OSAllCacheWriteBack 函数

函数原型	**void OSAllCacheWriteBack(void)**
函数功能	把所有已改变的扇区写回逻辑盘
入口参数	无
出口参数	无
说明	对一个文件执行完写操作后，先关闭文件，然后再调用该函数执行一次回写操作，才能确保信息写入到了介质中。

调用示例	OSFileWrite((uint8 *)Data, strlen(DataBase), fp1); // 写数据到文件
	OSFileClose(fp1); // 文件写入完毕后，关闭文件
	OSAllCacheWriteBack(); // 将缓冲区数据写回设备

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

表 4.22 OSFindFirst 函数

函数原型	**uint8 OSFindFirst(char *Name, char *Path)**
函数功能	在指定目录下，查看第一个文件（或目录）的名称
入口参数	Name 文件（或目录）名称保存缓冲区 Path 路径名称
出口参数	FIND_FILE 找到文件 FIND_DIR 找到目录 FDT_EOF 查找完毕 其它参考关于文件系统返回值的说明
说明	与 OSFindNext 函数配合，可列出一个目录下所有文件、目录的名称。

调用示例	char FileName[50];
	// 查找出根目录下第一个文件（或目录）的名称，并保存到缓冲区
	// FileName 内。至于这个名称是文件还是目录，可通过 FindData 判断。
	FindData = OSFindFirst(FileName,"A:\\");
	// 在根目录下，查找下一个文件（或目录）名称
	FindData = OSFindNext(FileName);

表 4.23 OSFindNext 函数

函数原型	**int32 OSFindNext(char *Name)**
函数功能	在指定目录下，查看下一个文件（或目录）的名称
入口参数	Name 文件（或目录）名称保存缓冲区
出口参数	FIND_FILE 找到文件 FIND_DIR 找到目录 FDT_EOF 查找完毕 其它参考关于文件系统返回值的说明
说明	与 OSFindFirst 函数配合，可列出一个目录下所有文件、目录的名称。

调用示例	char FileName[50];
	// 查找出根目录下第一个文件（或目录）的名称，并保存到缓冲区
	// FileName 内。至于这个名称是文件还是目录，可通过 FindData 判断。
	FindData = OSFindFirst(FileName,"A:\\");
	// 在根目录下，查找下一个文件（或目录）名称
	FindData = OSFindNext(FileName);

表 4.24 OSFormat 函数

函数原型	**uint8 OSFormat(char *DriveName, uint32 SecPerDisk, unsigned int BytsPerSec, unsigned int rsSecs)**
函数功能	格式化逻辑卷
入口参数	DriveName 包含驱动器名称的字符串驱动器 SecPerDisk 逻辑盘包含扇区数 BytsPerSec 扇区包含字节数目
出口参数	TRUE 成功 FALSE 失败
说明	该函数适用于除 M22A-N20I/C 之外的其他工控模块

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

表 4.25 Format 函数

函数原型	**uint8 Format(char *DriveName, uint32 SecPerDisk, uint16 BytsPerSec)**
函数功能	格式化逻辑卷
入口参数	DriveName 包含驱动器名称的字符串驱动器 SecPerDisk 逻辑盘包含扇区数 BytsPerSec 扇区包含字节数目
出口参数	TRUE 成功 FALSE 失败
说明	该函数只适用于 M22A-N20I/C 工控模块

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

5. 存储设备 API 函数说明

5.1 CF 卡驱动函数

CF 卡是一种大容量存储设备，采用闪存技术，不需要电池来维持存储的数据，成本低，兼容性好，目前广泛应该在数码相机、PDA、MP3、工控机等嵌入式系统中。

5.1.1 函数说明

MiniARM 嵌入式工控模块进行 CF 卡应用时需要配合使用文件系统相关函数，相关函数说明请参考文件系统 API 函数说明小节，其他函数如表 5.1~表 5.3 所示。

表 5.1 CF 卡相关驱动函数一览表

函数名称	功能描述
CFIDEInit	CF 卡 IDE 接口初始化
GetCFCommand	获取 CF 卡硬盘驱动程序地址

表 5.2 CFIDEInit 函数

函数原型	**void CFIDEInit(void)**
函数功能	CF 卡 IDE 接口初始化
入口参数	无
出口参数	无
调用示例	CFIDEInit(); // 初始化 CF 卡 IDE 接口

表 5.3 GetCFCommand 函数

函数原型	**void * GetCFCommand(void)**
函数功能	获取 CF 卡驱动程序地址
入口参数	无
出口参数	CF 卡驱动程序地址
调用示例	void *DiskCommand; DiskCommand = GetCFCommand(); // 获取 CF 卡驱动程序地址

5.1.2 示范例程

在 CF 卡中建立目录 Dir，然后在该目录下再建立文件 Test.txt，向文件中写入字符串“广州致远电子有限公司”，如

程序清单 5.1 所示。

文件建立成功后，用户可以在 PC 机上观察 CF 中的数据。

步骤 1：在 ADS 开发环境下添加 MiniARM 工程模板（如果已经添加，该步骤可省略）。

这一步很简单，在此处不作过多介绍。

步骤 2：在 ADS 开发环境下使用 MiniARM 工程模板建立工程—File_Test。

步骤 3：编写 ZLG/FS 应用程序。

在 main 函数中初始化文件系统，如程序清单 5.1 所示。

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

程序清单 5.1 main 函数—文件系统初始化

```
int main (void)
{
    void  *Command;
    uint8 CFSel;

#if OS_CRITICAL_METHOD == 3                    // Allocate storage for CPU status register */
    OS_CPU_SR    cpu_sr;
#endif

    TargetInit();                              // 初始化设备
    PinInit();
    ……

    File_init();                               // 初始化文件系统
    CFIDEInit();                               // CF 卡 IDE 接口初始化
    Command = GetCFCommand();                  // 获取 CF 卡驱动程序地址
    CFSel = 0;
    OSAddFileDriver(Command,&CFSel);           // 加载 CF 卡驱动,使用 CF 卡主模式

    ……
}
```

在任务 Task0 中完成对 CF 卡的操作，如程序清单 5.2 所示。

程序清单 5.2 任务 Task0——操作 CF 卡

```
/*********************************************************************************
** Function name:     TASK0
** Descriptions:      在 CF 卡中建立一个文件，并在该文件中写入：广州致远电子有限公司。
** Input:             无
** Output:            无
*********************************************************************************/
void TASK0(void *pdata)
{
    HANDLE  fp;
    char  WriteBuf[100];

    pdata = pdata;
    OSMakeDir("A:\\Dir");                  // 在根目录下建立一个目录 Dir          ①
    OSChangeDir("A:\\Dir");                // 更改当前路径为：A:\\Dir             ②
    fp = OSFileOpen("Test.txt","RW");      // 在目录 Dir 下建立文件:Test.txt      ③
    if (Not_Open_FILE != fp) {             // 打开文件成功
                                           // 向文件中写入信息：广州致远电子有限公司
        sprintf(WriteBuf,"广州致远电子有限公司");                                  ④
        OSFileWrite((void *)WriteBuf,strlen(WriteBuf),fp);                      ⑤
```


©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

```
            OSFileClose(fp);                    // 关闭文件                    ⑥
            OSAllCacheWriteBack();              // 将更新后的 Cache 回写到 CF 卡中   ⑦
        } else {
            while(1) {                          // 文件打开失败
                OSTimeDly(30);
            }
        }

        while (1) {
            OSTimeDly(OS_TICKS_PER_SEC);
        }
}
```

首先在 CF 中建立一个目录 Dir（如程序清单 5.2 中的①），在默认的情况下，系统的当前路径为根目录，那么现在就将当前路径更改到 Dir 处（程序清单 5.2 中的②）。接下来在 Dir 目录下建立一个文件 Test.txt（程序清单 5.2 中的③），如果文件创建成功，则向 Test.txt 中写入信息：广州致远电子有限公司（程序清单 5.2 中的④～⑥）。最后将 Cache 中的所有信息全部回写到 CF 卡中（程序清单 5.2 中的⑦）。

系统运行环境为：MiniARM 嵌入式工控模块＋M22A 评估板。将 CF 卡插入接口 CON7，跳线 JP12 短接（采用主模式），运行程序。然后取出 CF 卡，在 PC 上打开，可以观察到 Test.txt 文件中的信息。如图 5.1 所示。其中，CF 卡对应的盘符为 H，可见，在 CF 卡中有一级目录 Dir，同时在 Dir 目录下面只有一个文件 Test.txt，该文件中的信息为：广州致远电子有限公司。

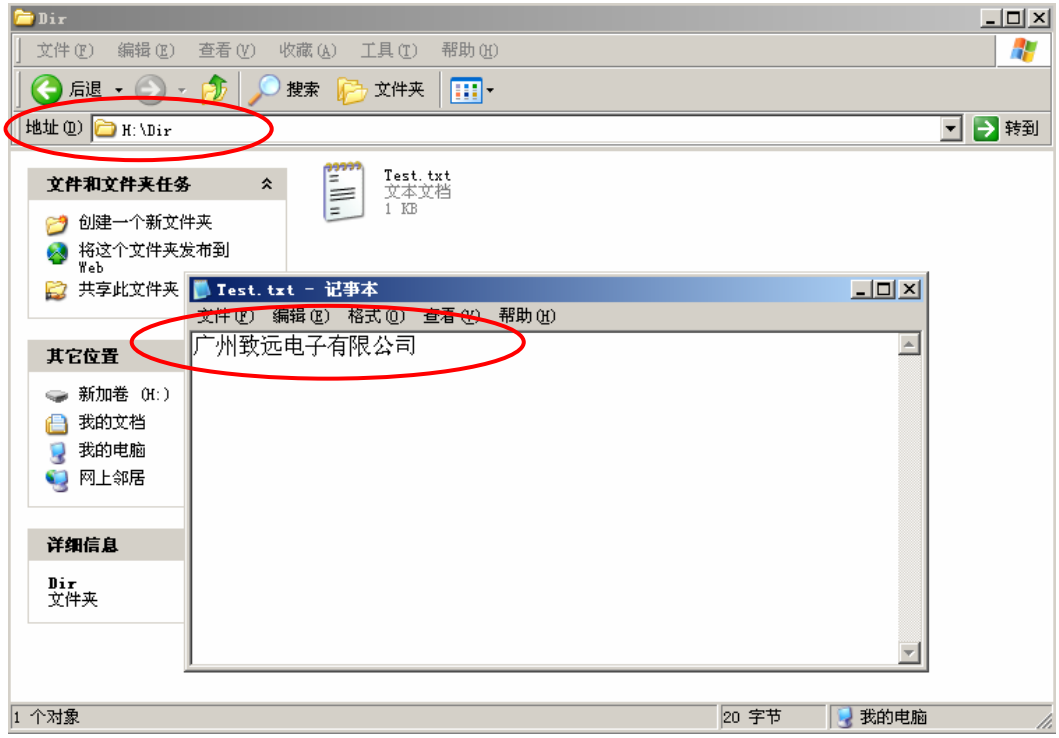

图 5.1 检查存储介质中的数据

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

5.2 U 盘驱动函数

MiniARM 嵌入式工控模块提供的 USB 主机接口主要用于接入 USB 存储设备，MiniARM 为用户提供了完整的 USB 大容量存储设备操作函数库，配合文件系统，即可完成对 USB 存储设备的文件操作。

由于 USB 接口具有即插即用和可热插拔的特性，MiniARM 嵌入式工控模块的驱动程序提供了 USB 设备存在与否的检测函数，建议用户程序也提供对这些特性的支持。

5.2.1 设备描述结构体

如程序清单 5.3 所示，大容量存储设备通过 MassDriver 结构体描述。

程序清单 5.3 海量存储设备结构体

```
typedef struct _MassDriver{
    uint8       DeID;                   // 设备 ID，按设备接入先后排序
    uint8       LunID;                  // 逻辑单元索引号
    hMedLUN *MediumPtr;                 // 驱动库内部参数,用户不需理会
}MassDriver;
```

设备 ID 由驱动程序按照设备接入先后自动分配 0、1……等序号，例如，同时使用 2 个 U 盘时，第一个接入的 U 盘 MassDriver.DeID= 0，第二个 U 盘 MassDriver.DeID= 1。

逻辑单元索引号用于识别读卡器中的不同设备，MassDriver. LunID 的序号是由读卡器生产厂商分配的，例如市面常见的读卡器 SD 卡 MassDriver. LunID= 0，CF 卡 MassDriver. LunID=1。但用户需要注意，MassDriver. LunID 与设备分区不一样，分区由文件系统识别，而 MassDriver. LunID 由驱动程序识别。

*MediumPtr 是驱动程序内部参数，无需用户理会。

5.2.2 函数说明

MiniARM嵌入式工控模块进行USB大容量存储设备应用时需要配合使用文件系统相关函数，相关函数说明请参考文件系统API函数说明小节，其它函数如表 5.4~表 5.8所示。

表 5.4 USB 相关驱动函数一览表

函数名称	功能描述
USB_Host_init()	USB 主机控制器初始化
GetUSBCommand()	获取 USB 控制器驱动程序函数地址
MassDriverIni()	大容量设备初始化
find_mass_device()	检测是否有大容量设备接入

表 5.5 USB_Host_init 函数

函数原型	**void USB_Host_init(void);**
函数功能	USB 主机初始化
入口参数	无
出口参数	无
调用示例	USB_Host_init(); ……

表 5.6 GetUSBCommand 函数

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

函数原型	**void * GetUSBCommand（void）;**
函数功能	获取 USB 控制器驱动程序函数地址
入口参数	无
出口参数	驱动程序地址指针
调用示例	void * USBCommand; …… USBCommand =GetUSBCommand(); //获取 USB 底层驱动程序地址

表 5.7 MassDriverIni 函数

函数原型	**void MassDriverIni(void);**
函数功能	大容量存储设备初始化
入口参数	无
出口参数	无
调用示例	MassDriverIni(); ……

表 5.8 find_mass_device 函数

函数原型	**device_instance find_mass_device(unsigned char DeviceIndex);**
函数功能	检测大容量设备是否存在
入口参数	DeviceIndex: 设备逻辑单元索引号;
出口参数	dvi_ptr_: 设备描述信息结构指针， 如果设备不存在，将返回空指针“NULL”

调用示例	if(find_mass_device(mass_driver.DeID) == NULL) // 等待 U 盘插入
	{
	OSTimeDly(10);
	continue;
	}

5.2.3 示范例程

使用MiniARM嵌入式工控模块进行U盘读写操作的范例如程序清单 5.4所示。程序运行后，如果插入U盘，将在U盘内创建目录和文件，并写入数据。操作成功后，可以在PC机上查看运行结果。

程序清单 5.4 U 盘文件操作范例

```
char  DIR[]="A:\\MiniARM";
char  FNAME[] = "ReadMe.TXT";
char  const WritFileData[]=”It is a test!”;

void UDiskFSDemoFunc(void)
{
    HANDLE   FHandle;                          // 文件句柄
    void *USBCommand;                          // 驱动程序地址指针
     MassDriver mass_driver;
```


©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

```
    USB_Host_init();                                    // 初始化
    MassDriverIni();
    File_init();
    mass_driver.DeID = 0;                               // 设备 ID
    mass_driver.LunID = 0;                              // 逻辑单元索引号
    USBCommand = GetUSBCommand();                       // 获取 USB 底层驱动程序地址
    while (1) {
        if (find_mass_device(mass_driver.DeID) == NULL) {   // 等待 U 盘插入
            OSTimeDly(10);
            continue;
        }
        OSAddFileDriver(USBCommand, &mass_driver);      // 安装文件系统驱动
        if (mass_driver.MediumPtr == NULL) {
            OSRemoveFileDriver(USBCommand, &mass_driver);
            OSTimeDly(10);
            continue;
        }
        OSMakeDir(DIR);                                 // 创建目录
        OSChangeDir(DIR);                               // 更改当前路径
        FHandle = OSFileOpen(FNAME, "RW");              // 在当前路径打开或创建文件
        if (Not_Open_FILE != FHandle) {                 // 打开文件成功
            OSFileSeek(FHandle, 0, SEEK_END);           // 将指针移到文件末尾
            OSFileWrite((uint8 *)WritFileData,
                sizeof(WritFileData),FHandle);          // 写数据入文件
            OSFileClose(FHandle);                       // 关闭文件
            OSAllCacheWriteBack();                      // 将缓冲区数据写回设备
            GpioSet(BUZZER);
            OSTimeDly(30);
            GpioClr(BUZZER);                            // 蜂鸣器蜂鸣
        } else {
            GpioSet(BUZZER);                            // 文件打开失败
        }
        while (find_mass_device(mass_driver.DeID) != NULL) {
            OSTimeDly(10);                              // 等待 U 盘拔除
        }
        OSRemoveFileDriver(USBCommand, &mass_driver);   // 删除驱动
        GpioClr (BUZZER);
    }
    while(1) {
        OSTimeDly(OS_TICKS_PER_SEC) ;
    }
}
```


©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

5.3 电子盘驱动函数

MiniARM 嵌入式工控模块集成有 256MB 板载电子盘。用户可以通过固化的文件系统将大容量的数据储存到电子盘，无需再使用 CF 卡或者 IDE 硬盘，使产品获得出色的稳定性和温度特性。

由于电子盘已经集成于模块内部，所以无需检测其是否接入。使用电子盘进行文件读写操作前，只需初始化相关的设备和参数即可。需要注意的是，同 USB 和 CF 卡接口一样，电子盘的使用需要配合文件系统，才能完成文件操作。

5.3.1 函数说明

由于电子盘所使用的 NAND Flash 芯片扇区大小有 512Byte 和 2Kbyte 之分，对应需要使用不同的驱动程序。常用的三星芯片中，早期的芯片使用的是 512Byte 大小的扇区结构，从 128MByte 容量开始，扇区大小都是 2Kbyte。

如果您使用的是集成 16M 电子盘模块，应该配合 GetFFSCommand()、Get16MDriverInfo ()函数加载驱动。如果是 256M 电子盘配置，则使用 GetLFFSCommand()、Get256MDriverInfo() 函数加载驱动。

表 5.9 电子盘相关驱动函数一览表

函数名称	功能描述
GetFFSCommand()	获取电子盘驱动程序函数地址（512Byte 大小扇区）
Get16MDriverInfo ()	获取电子盘硬件结构信息（16M 芯片）
GetLFFSCommand()	获取电子盘驱动程序函数地址（2KByte 大小扇区）
Get256MDriverInfo()	获取电子盘硬件结构信息（256M 芯片）

表 5.10 GetFFSCommand 函数

函数原型	**void *GetFFSCommand(void)**
函数功能	获取 NandFlash 驱动程序地址(512bytes 扇区类型)
入口参数	无
出口参数	NAND Flash 驱动程序地址
调用示例	void *DiskCommand; DiskCommand = GetFFSCommand (); //获取 NandFlash 驱动程序地址
说明	该函数只适用于 512bytes 每扇区结构的 NAND Flash，如果是 2Kbytes 扇区结构的 NAND Flash，需要使用 GetLFFSCommand ()函数

表 5.11 GetLFFSCommand 函数

函数原型	**void * GetLFFSCommand (void)**
函数功能	获取 NandFlash 驱动程序地址(2Kbytes 扇区类型)
入口参数	无
出口参数	NAND Flash 驱动程序地址
说明	该函数只适用于 2Kbytes 每扇区结构的 NandFlash

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

表 5.12 Get16MDriverInfo 函数

函数原型	**void * Get16MDriverInfo (void)**
函数功能	获取 NandFlash 硬件结构信息(512bytes 扇区类型)
入口参数	无
出口参数	NAND Flash 芯片信息
调用示例	void *DiskInfo; DiskInfo = Get16MDriverInfo (); //获取 NandFlash 芯片信息
说明	配合 GetFFSCommand()使用

表 5.13 Get256MDriverInfo 函数

函数原型	**LFFSDisk * Get256MDriverInfo (void)**
函数功能	获取 NandFlash 硬件结构信息(2Kytes 扇区类型)
入口参数	无
出口参数	NAND Flash 芯片信息
调用示例	LFFSDisk *LDiskInfo LDiskInfo = Get256MDriverInfo (); //获取 NandFlash 芯片信息
说明	配合 GetLFFSCommand()使用

5.3.2 示范例程

使用板载电子盘进行文件读写操作的范例如程序清单 5.5所示。程序运行后，在电子盘内创建目录和文件，并写入数据。

程序清单 5.5 电子盘文件操作范例

```
char  const WritFileData[]=”It is a test!”;

void NandFFSDemoFunc(void)
{
      HANDLE  FHandle;
      uint32       filesize = 0;                                    // 文件大小

      File_init();
      OSAddFileDriver(GetLFFSCommand(),Get256MDriverInfo ());      // 256M 电子盘

#if 1
     OSFormat("a:", 0, 0, 32);                                     // 格式化 NAND Flash
#endif

      FHandle = OSFileOpen("A:\\test.txt", "RW");                  // 打开或创建文件
      if (Not_Open_FILE != FHandle) {                              // 打开文件成功
            OSFileSeek(FHandle,0, SEEK_END);                       // 将指针移到文件末尾
            filesize = OSFileWrite((uint8 *)WritFileData,
                        sizeof(WritFileData),FHandle);             // 写数据入文件
```


©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

```
            OSFileClose(FHandle);                                   // 关闭文件
            OSAllCacheWriteBack();                                  // 将缓冲区数据写回设备

            GpioSet(BUZZER);                                        // 蜂鸣器短鸣
            OSTimeDly(20);
            GpioClr(BUZZER);
        } else {
            GpioSet(BUZZER);                                        // 文件打开失败
        }

        while (1) {
            OSTimeDly(OS_TICKS_PER_SEC);
        }
}
```


©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

6. 免责声明

开发预备知识

MiniARM® M22A 系列产品将提供尽可能全面的开发模板、驱动程序及其应用说明文档以方便用户使用，但 MiniARM® M22A 系列产品不是教学开发平台。对于需要熟悉 ARM7 体系结构，LPC2200 系列微控制器特性及其 ADS 开发环境的用户，建议同时购买我公司 SmartARM2200 或 EasyARM2200 教学开发平台。

LPC2000 系列微控制器

建议用户在 NXP 半导体主页（http://www.nxp.com）上获取最新勘误表并仔细阅读。广州致远电子有限公司对 LPC2200 系列微控制器无论是已知的还是潜在的设计缺陷不负任何责任。

修改文档的权利

广州致远电子有限公司保留任何时候在不事先声明的情况下对 MiniARM® M22A 系列产品相关文档的修改的权力。

ESD 静电放电保护

MiniARM® M22A 系列产品部分元器件内置 ESD 保护电路，但依然建议用户在设计底板时提供 ESD 保护措施，特别是电源与 I/O 设计，以保证产品的稳定运行。安装 MiniARM® M22A 系列产品时，请先将积累在身体上的静电释放，例如佩戴可靠接地的静电环，触摸接入大地的自来水管等。

©2008 Guangzhou ZHIYUAN Electronics CO., LTD.

公　　司：广州致远电子有限公司 嵌入式系统事业部
地　　址：广州市天河区车陂路黄洲工业区二栋四楼（研发部）
邮　　编：510660
网　　址：www.embedtools.com
销售电话：+86 (020) 2264-4249
技术支持：+86 (020) 2887-2684
传　　真：+86 (020) 3860-1859