U 盘和 SD 卡文件管理芯片 CH376 的评估板说明及应用参考

版本： 1

http://wch.cn

1、概述

CH376 评估板包含 CH376S 芯片和辅助元器件，不含单片机，对外预留了 8 位并口、SPI 接口、异步串口以及电源端口等，用于连接其它单片机主板，并由单片机主板控制 CH376 进行功能评估。

在单片机的控制下，可以用于演示 CH376 的 U 盘和 SD 卡文件管理功能、USB-HOST 主机接口功能和 USB 设备接口功能。可以读写 U 盘（闪盘、USB 闪存盘、USB 外置硬盘、USB 读卡器等）和 SD 卡（标准容量 SD 卡和高容量 HC-SD 卡以及协议兼容的 MMC 卡和 TF 卡）；可以作为 USB 设备与计算机相连接，例如自行设计的专用 U 盘等；通过相关程序，还可以操作其它 USB 设备，例如 USB 打印机、USB 键盘、USB 鼠标等，或者与另一块 CH37X 评估板进行对连，互传数据。

本文中涉及 MCS51 单片机的大多数实验和例子程序都是在 CH375 评估板上进行的，原则上，应该使用排线等将 CH376 评估板连接到 CH375 评估板的对外端口 P6。只不过，由于 CH376S 芯片的引脚基本兼容 CH375 芯片，所以只需要将 CH375 评估板上的 CH375 换成 CH376 即可进行 CH376 芯片的功能评估，而并不需要通过 P6 端口连接 CH376 评估板。实际上，简单的做法就是将 CH375 的 DIP28 转换板换成 CH376S 的 DIP28 转换板，采用 CH376 的 DIP28 转换板与通过 P6 端口外接 CH376 评估板的唯一区别就是 I/O 地址不同。有关 CH375 评估板的硬件请参考 CH375 评估板的资料 CH375EVT.PDF，网上的压缩文件是 CH375EVT.ZIP。

2、评估板的硬件

评估板的原理图和 PCB 请参考 CH376SCH.PDF 文档。下面是元器件说明。

评估板中的主要器件 U1 是 CH376S 芯片，但是图中有些信号是以 CH375 或者 CH374 命名的。

晶体 X1 为标准 12MHz，USB 主机比 USB 设备要求更高的频率精确度，X1 的误差要求小于 0.4‰，普通的 12MHz 晶体基本上可以满足要求。强烈建议缩短相关引线的长度，以减少干扰。

电容 C4 用于内部电源节点退耦从而降低 USB 传输过程中的 EMI，容量为 4700pF 到 0.1uF，可以选用普通的 103 贴片电容 0.01uF。

P4 是 USB 端口，既可以用于 USB HOST 主机方式，也可以用于 USB DEVICE 设备方式，电阻 R1 用于限制输出给外部 USB 设备的电流，避免在 U 盘等 USB 设备刚插入时造成电源电压的短时间下降，甚至引起 CH376 或者单片机非正常复位或者内部 RAM 数据错误。如果是 USB 外置硬盘，那么应该将电阻 R1 换成直流电阻较小的电感，或者另外用一组 5V 电源直接提供更大的工作电流（500mA 以上）给外置硬盘。另外，USB-HOST 插座的电源退耦电容 C9 的容量不能太小，容量大些（应该大于 100μF）可以减少在 USB 设备刚插入时的电源电压的波动。

P5 是 SD 卡插座，可以接触标准尺寸的 SD 卡，其它规格的 SD 卡可以需要另加转换座。电阻 R3 用于限制输出给外部 SD 卡的电流，避免在 SD 卡刚插入时造成电源电压的短时间下降。

P1 是 8 位并口的信号端口，用于连接单片机的并口，并口的必要信号包括 D0-D7、A0、RD#、WR#、CS#以及 GND，而 INT#是可选的。

P2 是 SPI 串口的信号端口，用于连接单片机的 SPI 接口，SPI 的必要信号包括 SCS、SCK、SDI、SDO 以及 GND，而 INT#是可选的。

P3 是异步串口的信号端口，用于连接单片机的异步串口，异步串口的必要信号包括 RXD、TXD 以及 GND，而 INT#是可选的。P3 同时提供了 SD 卡的写保护 SDWP 和插拔状态 SDINSERT 信号线。

上述 P1、P2、P3 通讯端口，还可以从外部向本评估板提供 5V 电源，以及向 CH376 的 RSTI 引脚提供可选的硬件复位信号，如果实际产品电路中有μP 监控电路，那么应该为 CH376 和单片机提供同一个复位信号。注意，应该尽量缩短本评估板与单片机之间的信号线的长度，最长不能超过 20cm，否则需要使用一个信号间隔一个地线的专用排线。

J3 用于选择 CH376 芯片工作电压，短接 1-2 脚时为 5V 电压，短接 2-3 脚时为 3.3V 电压。默认为 5V 电压，但当单片机的工作电压等于或者低于 3.3V 时，可以为 CH376 选择 3.3V 电压。当 CH376 芯片的工作电压为 5V 时，J2 必须断开，当 CH376 芯片的工作电压为 3.3V 时，J2 必须短接。

J1、J5 和 J6 用于在上电或者硬件复位后选择 CH376 与单片机的通讯接口：

如果 J1 短接、J5 断开、J6 断开，那么是 8 位并口；

如果 J1 断开、J5 短接、J6 短接，那么是 SPI 接口；

如果 J1 断开、J5 断开、J6 断开，那么是异步串口。

有些例子程序可能会用单片机的串口输出调试状态信息，如果需要显示这些监控信息，可以由将单片机串口经过 RS232 电平转换后连接到计算机使用串口监控/调试工具软件查看。如果使用 CH375 评估板，那么可以将 J2 连接到计算机串口；如果计算机没有串口，或者串口已经被其它设备占用，那么可以由 USB 转串口芯片 CH341 提供仿真串口。

CH375 评估板内部器件工作于 5V 电源电压时，必须加上电阻 R0 并去掉 3.3V 稳压器 D4，工作于 3.3V 电源电压时，必须加上稳压器 D4 并去掉电阻 R0。默认是 5V 电源。

3、示例程序

产品光盘或者网站上提供了 CH376 的一些 C 语言示例程序，可以用作设计参考，另外，还可以参考 CH376 的 U 盘文件读写模块中的多个 C 和 ASM 示例程序。这些示例大多数是以 MCS51-C 语言或者 MCS51-ASM 语言编写，将硬件相关部分局部修改后，基本上可以适用于其它类型的单片机或者 DSP。有关 USB 设备方式的应用可以参考 CH372 评估板资料中的说明。

如果在 CH375 评估板中做 CH376 的 DIP 转换板测试，那么 CH376 的端口地址是（可用其它地址）：

```
unsigned char volatile xdata CH376_CMD_PORT _at_ 0xBDF1; /* CH376 命令端口的 I/O 地址 */
unsigned char volatile xdata CH376_DAT_PORT _at_ 0xBCF0; /* CH376 数据端口的 I/O 地址 */
#define CH376_INT_WIRE  INT0        /* 连接 CH376 的 INT#引脚，用于查询中断状态，可选 */
```

如果在 CH375 评估板中做 CH376 评估板的测试，那么 CH376 的端口地址是（也可以用其它地址）：

```
unsigned char volatile xdata CH376_CMD_PORT _at_ 0xEDF1; /* CH376 命令端口的 I/O 地址 */
unsigned char volatile xdata CH376_DAT_PORT _at_ 0xECF0; /* CH376 数据端口的 I/O 地址 */
#define CH376_INT_WIRE  INT1        /* 连接 CH376 的 INT#引脚，用于查询中断状态，可选 */
```

以下是硬件总线 8 位并口连接方式的 I/O 子程序：

```
void xWrite376Cmd( unsigned char cmd ) {  /* 向命令端口写入命令码，周期不小于 1.5uS */
    CH376_CMD_PORT=cmd;
    Delay1.5uS( );
}
void xWrite376Data( unsigned char dat ) {  /* 向数据端口写入数据，周期不小于 0.6uS */
    CH376_DAT_PORT=dat;  /* 标准 MCS51 单片机指令较慢，无需延时 */
}
unsigned char xRead376Data ( void ) {  /* 从数据端口读出数据，周期不小于 0.6uS */
    return( CH376_DAT_PORT );  /* 标准 MCS51 单片机指令较慢，无需延时 */
}
```

3.1. 读写 U 盘或者 SD 卡

这种方式下，CH376 会分析 U 盘或者 SD 卡中的文件系统，并按照其格式进行数据读写。写入数据后的 U 盘插到计算机中，计算机可以直接看到相应的文件，并可以直接读写其中的数据。

详细的例子可以参考 CH376/EVT 目录下的各个 EXAM 程序，在第 4、5、6 节有更详细的说明。

3.2. 操作其它 USB 设备

被操作的 USB 设备应该支持 USB1.1 或者 USB2.0 的 12Mbps 全速或者 1.5Mbps 低速。

CH376 内置了常用的 USB 底层固件程序，并且能够自动检测 USB 设备是否连接，当 USB 设备连接后，建议先进行 USB 总线复位，然后读取 USB 描述符、设置 USB 地址、设置 USB 配置、分析 USB 设备等，对于不同的 USB 设备，其操作流程及方式可能不同。

CH376 基本兼容 CH375，详细的例子可以参考 CH375/EVT/PUB/MCS51C/MISCELL 目录下的程序。

4、U 盘或 SD 卡文件管理

很多数码产品以及单片机系统都需要存储器，当前，U 盘和 SD 卡（含闪盘、USB 闪存盘、USB 移动硬盘、HC-SD 卡、TF 卡等，下同）已经成为很常用的移动存储设备，其价格仅比相同容量的闪存略高，而远比闪存易于采购和易于携带，并且 U 盘和 SD 卡的规格通用，具有多种容量可供选用。所以，数码产品以及单片机系统可以直接采用 U 盘或者 SD 卡作为大容量的移动存储器。

CH376 是 U 盘和 SD 卡文件管理控制芯片，能够存取 U 盘或者 SD 卡中的文件，支持 FAT32、FAT16 和 FAT12，符合 WINDOWS 的文件系统格式，方便电子产品中的单片机/DSP/SCM/CPU 等与使用 WINDOWS 操作系统的个人计算机交换数据。

4.1. 文件应用基础

U 盘（或者 SD 卡，下同）提供了若干个物理扇区用于数据存储，每个扇区大小通常是 512 字节。由于计算机通常将 U 盘中的物理扇区组织为 FAT 文件系统，为了方便单片机通过 U 盘或者 SD 卡与计算机之间交换数据，单片机也应该在 FAT 规范下通过文件的形式存取 U 盘中的数据。

一个 U 盘中可以有若干个文件，每个文件都是一组数据的集合，以文件名区分和识别。实际文件数据的存放可能不是连续的，而是通过一组“指针”链接的多个块（也就是分配单元或者簇），从而能够根据需要随时增大文件长度以容纳更多数据。目录（文件夹）是为了便于分类管理，管理者可以人为指定将多个文件归档在一起，例如 2004 年的文件归到一个目录（文件夹）中。

在 FAT 文件系统中，磁盘容量以簇为基本单位进行分配，而簇的大小总是扇区的倍数，所以文件的占用空间总是簇的倍数，也是扇区的倍数。虽然文件占用的空间是簇或者扇区的倍数，但是在实际应用中，保存在文件中的有效数据的长度却不一定是扇区的倍数，所以 FAT 文件系统在文件目录信息 FAT_DIR_INFO 中专门记录了当前文件中有效数据的长度，即有效数据的字节数，也就是通常所说的文件长度，文件长度总是小于或者等于文件占用的空间。

在对文件写入数据后，如果是覆盖了原数据，那么文件长度可能不发生变化，当超过原文件长度后，变为追加数据，那么文件长度应该发生变化（增大）。如果向文件追加数据后，没有修改文件目录信息中的文件长度，那么 FAT 文件系统会认为超过文件长度的数据是无效的，正常情况下，计算机无法读出超过文件长度的数据，虽然数据实际存在。

如果数据量少或者数据不连续，那么可以在每次追加数据后立即更新文件目录信息中的文件长度，但是，如果数据量大并且需要连续写入数据，立即更新文件目录信息会降低效率，并且频繁修改文件目录信息也会缩短 U 盘中闪存的使用寿命（因为闪存只能进行有限次擦写），所以在这种情况下，应该在连续写入多组数据后再更新一次文件目录信息中的文件长度，或者一直等到关闭文件时再更新文件长度，CMD_FILE_CLOSE 命令可以将内存中的文件长度刷新到 U 盘文件的文件目录信息中。

虽然 CH376 最大支持 1GB 的单个文件，但是为了提高效率，建议单个文件的长度不要超过 100MB，通常在几 KB 到几 MB 范围是比较正常的，数据较多时可以分多个目录，分多个文件存储。

4.2. 存取模式

CH376 对 U 盘文件的读写方式分为两种：扇区模式和字节模式；CH376 对 SD 卡文件的读写方式只有一种是字节模式。

扇区模式下，以扇区（每扇区通常是 512 字节）为基本单位对 U 盘文件进行读写，所以读写速度略快，但是通常情况下需要额外的文件数据缓冲区，额外的文件数据缓冲区必须是扇区长度 512 的整数倍，所以适用于 RAM 多、数据量大、频繁读写数据的单片机系统。扇区读写的子程序主要有扇区读 CH376SecRead 和扇区写 CH376SecWrite。

字节模式下，以字节为基本单位对 U 盘文件进行读写，少则 1 字节，多则 65535 字节，读写速度略慢，但是不需要额外的文件数据缓冲区，使用方便，适用于 RAM 少（从几字节到几十 K 都可以）、数据量小或者数据零碎、不经常读写数据的单片机系统。但是，因为闪存只能进行有限次擦写，如果频繁地向 U 盘写入零碎的数据，可能会缩短 U 盘中闪存的使用寿命。字节读写的子程序主要有字节读 CH376ByteRead 和字节写 CH376ByteWrite 及移动文件指针 CH376ByteLocate。

4.3. 应用范例的文件结构

有关 U 盘和 SD 卡文件读写的例子主要由以下文件组成：

① 硬件抽象层，即 I/O 接口子程序，由单片机与 CH376 芯片之间的通讯接口方式决定

HAL.H 硬件抽象层头文件

HAL_BASE.C 硬件抽象层基本子程序，包括延时子程序，需要根据单片机实际速度修改

I/O 接口子程序，必须根据实际的通讯方式选择以下其中一种并按实际硬件和参数进行修改：

PARA_HW.C 是硬件总线 8 位并口连接方式；

PARA_SW.C 是软件模拟 8 位并口连接方式；

SPI_HW.C 是硬件 SPI 接口连接方式；

SPI_SW.C 是软件模拟 SPI 接口连接方式；

UART_HW.C 是硬件异步串口连接方式，支持波特率从 9600bps 到 3Mbps。

② 文件系统层，将常用命令进行打包，提供了常用的文件管理子程序和一些不太常用的子程序

FILE_SYS.H 文件系统层头文件，子程序声明等；

FILE_SYS.C 文件系统层子程序，子程序源程序等，在第 5 节有详细说明。

对于 MCS51 单片机，可以使用针对其优化过的 FILE_SYS_C51.H 和 FILE_SYS_C51.C 文件。

为了节约单片机的程序 ROM 空间和数据 RAM 空间，默认情况下，不太常用的子程序会被禁止掉，需要使用时，可以在 include “FILE_SYS.H”和 include “FILE_SYS.C”之间定义以下宏：

定义 NO_DEFAULT_CH376_INT 用于禁止默认的 Wait376Interrupt 子程序，禁止后，应用程序必须自行定义一个同名子程序；

定义 DEF_INT_TIMEOUT 用于设置默认的 Wait376Interrupt 子程序中的等待中断的超时时间/循环计数值，0 则不检查超时而一直等待；

定义 EN_DIR_CREATE 用于提供新建多级子目录的子程序，默认是不提供；

定义 EN_DISK_QUERY 用于提供磁盘容量查询和剩余空间查询的子程序，默认是不提供；

定义 EN_SECTOR_ACCESS 用于提供以扇区为单位读写文件的子程序，默认是不提供；

定义 EN_LONG_NAME 用于提供支持长文件名的子程序，默认是不提供，使用长文件名子程序必须先定义全局缓冲区 GlobalBuf，长度不小于 64 字节，可以与其它子程序共用。

有些子程序要求输入文件名参数，有三种：

name 参数是指短文件名，通常是根目录下的文件（含有根目录符）或者当前目录下的文件，不能含有路径分隔符，总长度不超过 1+8+1+3+1 字节，最末字节为数据 0；

PathName 参数是指全路径的短文件名，包括根目录符、多级子目录及路径分隔符、文件名或者目录名；

LongName 参数是指长文件名，以 UNICODE 小端顺序编码，以两个 0 字节结束。

③ 辅助调试子程序，用于打印输出调试信息，以便检查程序运行结果，正式产品中必须删除之

DEBUG.H 辅助调试子程序头文件；

DEBUG.C 辅助调试子程序源程序等。

④ 应用程序主程序，针对一些常见的实际应用提供程序范例，分别位于 EXAM??各个子目录中。不同单片机的 C 语言示例程序基本通用，尤其是 main 主程序基本上适用于所有单片机，只需要修改硬件 I/O 相关部分，重新编译和链接就可以使用。在第 6 节有详细说明。

4.4. 文件读写的参考步骤

4.4.1. 初始化，进行任何一项文件操作之前的必要步骤

① 调用 mInitCH376Host 初始化，进入 USB-HOST 工作方式或者 SD 卡主机工作方式（模式 3）

② 等待 U 盘或者 SD 卡连接，U 盘可以由 CH376 自动检测并产生中断通知，或者由单片机调用子程序 CH376DiskConnect 定期查询，SD 卡必须由单片机自行检测

③ 调用 CH376DiskMount，初始化 U 盘或者 SD 卡，并测试磁盘是否就绪，失败后可以重试最多 5 次

④ 上述步骤只需执行一次，除非 U 盘或者 SD 卡断开后重新连接，那么必须回到步骤②

4.4.2. 顺序读文件
① 调用 CH376FileOpenPath，打开文件
② 多次调用 CH376ByteRead，读取数据
③ 调用 CH376FileClose，关闭文件，可选操作

4.4.3. 顺序改写文件（覆盖原数据，超过原文件长度后转变为追加数据）
① 调用 CH376FileOpenPath，打开文件
② 多次调用 CH376ByteWrite，写入数据
③ 调用 CH376FileClose，参数是 TRUE，关闭文件并允许自动更新文件长度

4.4.4. 向已有文件追加数据
① 调用 CH376FileOpenPath，打开文件
② 调用 CH376ByteLocate，参数是 0xFFFFFFFF，移动文件指针到文件末尾
③ 多次调用 CH376ByteWrite，写入数据
④ 调用 CH376FileClose，参数是 TRUE，关闭文件并允许自动更新文件长度

4.4.5. 新建文件并写入数据
① 调用 CH376FileCreatePath，新建文件
② 多次调用 CH376ByteWrite，写入数据
③ 调用 CH376FileClose，参数是 TRUE，关闭文件并允许自动更新文件长度

4.4.6. 先读文件再改写文件
① 调用 CH376FileOpenPath，打开文件
② 多次调用 CH376ByteRead，读取数据
③ 调用 CH376ByteLocate，参数是 0，移动文件指针到文件头部
④ 多次调用 CH376ByteWrite，写入数据
⑤ 调用 CH376FileClose，参数是 TRUE，关闭文件并允许自动更新文件长度

4.4.7. 如果文件已经存在则追加数据，如果文件不存在则新建文件再写入数据
① 调用 CH376FileOpenPath，打开文件，如果返回 ERR_MISS_FILE 说明文件不存在，那么转步骤③
② 调用 CH376ByteLocate，参数是 0xFFFFFFFF，移动文件指针到文件末尾，然后转步骤④
③ 调用 CH376FileCreatePath，新建文件
④ 多次调用 CH376ByteWrite，写入数据
⑤ 调用 CH376FileClose，参数是 TRUE，关闭文件并允许自动更新文件长度

4.4.8. 定期采集数据（数据量较小时参考 EXAM7，数据量较大时参考 EXAM8）
① 采集之前，调用 CH376FileCreatePath，新建文件
② 采集数据，转换为相应的格式，例如二进制数据、字符串等
③ 调用 CH376ByteWrite，写入数据，一次写不完，可以分多次写入
④ 如果要等很长时间才有下一组数据，为了避免在此期间发生断电、U 盘拔出等事件，导致文件长度不正确，可以用 CH376ByteWrite 写入 0 长度的空数据，强制更新文件长度
⑤ 如果整个采集过程结束，或者文件已经太大，那么转到步骤⑥，否则转到步骤②
⑥ 调用 CH376FileClose，参数是 TRUE，关闭文件并允许自动更新文件长度
⑦ 如果是因为文件已经太大的原因，那么转到步骤①，新建另一个文件名不同的新文件后继续

4.4.9. 修改文件名、文件日期/时间、文件长度等文件目录信息
① 调用 CH376FileOpenPath，打开文件
② 调用 CH376DirInfoRead，将文件目录信息读入内存
③ 调用 CH376ReadBlock 读出原文件目录信息
④ 调用 CH376DirInfoRead，将文件目录信息读入内存

⑤ 调用 CH376WriteOfsBlock 写入新的文件目录信息
⑥ 调用 CH376DirInfoSave，保存文件目录信息
⑦ 调用 CH376FileClose，参数是 FALSE，关闭文件并禁止自动更新文件长度，可选操作

4.4.10. 创建子目录（文件夹）
① 调用 CH376DirCreatePath，新建子目录（文件夹）
② 调用 CH376FileClose，参数是 FALSE，关闭目录并禁止自动更新文件长度

4.4.11. 新建具有小写文件名和长文件名的文件
① 根据长文件名生成短文件名，将短文件名组合成全路径
② 调用 CH376CreateLongName，新建具有长文件名的文件
③ 文件创建成功，下面可以按无长文件名的文件处理，例如调用 CH376FileOpenPath 打开

4.4.12. 获取文件的长文件名
① 提供短文件名，或者通过枚举等方法获得短文件名
② 调用 CH376GetLongName，根据短文件名获得相应的长文件名
③ 下面可以按无长文件名的文件处理，例如调用 CH376FileOpenPath 打开

4.4.13. 搜索和枚举文件名，全盘枚举所有文件，请参考有关 EXAM13 例子中的说明

4.4.14. 主从切换，与计算机通讯，读写 U 盘或者 SD 卡文件，请参考有关 EXAM0 例子中的说明

5、文件系统子程序说明

5.1. FILE_SYS 子程序

以下是 FILE_SYS.C 文件中主要子程序的说明

```
mInitCH376Host( );        /* 初始化 CH376 芯片 */
```
在开机后或者复位 CH376 之后，应该重新初始化 CH376，初始化时会简单测试通讯接口。

```
CH376ReadBlock( buf );  /* 从当前 CH376 内部缓冲区读取数据块,返回长度 */
```
能够产生中断的操作执行完成后，如果有结果数据，那么可以调用 CH376ReadBlock 读取。

```
CH376DiskConnect( );      /* 检查 U 盘是否连接,不支持 SD 卡 */
```
检查 U 盘是否连接，而 SD 卡必须由单片机直接查询 SD 卡座的插拔状态引脚。示例：
```
s=CH376DiskConnect( ); /* 查询 U 盘是否连接，返回 USB_INT_SUCCESS 则说明当前已连接 */
if ( s==USB_INT_SUCCESS ) /* 已经连接 */
else if ( s==ERR_DISK_DISCON ) /* 已经断开 */
```

```
CH376DiskMount( );        /* 初始化磁盘并测试磁盘是否就绪 */
```
初始化并检查 U 盘或者 SD 卡是否准备好，准备好后才能进行文件读写。示例：
```
s=CH376DiskMount( ); /* 查询 U 盘或 SD 卡是否准备好，有些 U 盘可能需多次调用才能成功 */
if ( s!=USB_INT_SUCCESS ) /* 还未准备好 */ else /* 准备好了，可以读写数据 */
```

```
CH376FileOpenPath( PathName );  /* 打开多级目录下的文件或者目录(文件夹) */
CH376FileOpen( name );          /* 在根目录或者当前目录下打开文件或者目录(文件夹) */
```
调用前，应该在 PathName 或者 name 中提供文件名，包括完整的路径名，例如
```
\WINDOWS/SYSTEM/MYLIB.DLL      只能用 CH376FileOpenPath
\TODAY1.TXT                    可以用 CH376FileOpenPath 或者 CH376FileOpen
```

\YEAR2004\MONTH05\DATE18.DAT 只能用 CH376FileOpenPath

路径名和文件名的格式与 DOS 文件名格式相同，但是不含盘符和冒号，左斜杠与右斜杠等效，所有字符必须是大写，不能使用通配符，文件名长度不超过 11 个字符，其中主文件名不超过 8 个字符，扩展名不超过 3 个字符，如果有扩展名，则用小数点与主文件名隔开。如果单片机 RAM 有限，那么子目录的深度不宜超过 3 级，如果是 RAM 足够，那么路径名没有长度限制。示例：

```
strcpy( PathName, "\\YEAR2004\\CH376HFT.C" );
CH376FileOpenPath( PathName );  /* 打开文件 */
```

如果路径名太长，那么可以分多次逐级打开，首先打开子目录，直到最后再打开文件，其中，首次打开必须是从根目录开始，所以路径名首字符必须是斜杠，以后接着前级再打开时的首字符必须不是斜杠。示例：打开文件"/YEAR2004/MONTH05/DATE18/HOUR08/ADC.TXT"

```
strcpy( PathName, "/YEAR2004/MONTH05/DATE18" ); /* 目录名 */
s=CH376FileOpenPath( PathName ); /* 因为路径名太长，所以分两次打开，先打开前 3 级 */
if ( s==USB_INT_SUCCESS ) {  /* 前 3 级子目录成功打开，下面接着打开下级目录及文件 */
  strcpy( PathName, "HOUR08/ADC.TXT" ); /* 首字符不是斜杠 */
  s=CH376FileOpenPath( PathName ); /* 打开第 4 级子目录和文件 */
}
if ( s!=USB_INT_SUCCESS ) /* 出错 */ else /* 成功打开文件 */
```

如果文件名参数为"/"则可以打开根目录，从而便于自行处理长文件名，用完后必须关闭。如果打开的是子目录，那么文件长度总是 0xFFFFFFFF，否则为真正的文件长度。

CH376FileOpen(name); /* 枚举文件 */

以通配符*代替需要查询的文件名中的全部或者部分字符（通配符*后面不能再有字符），就可以枚举目录下的所有文件或者子目录，枚举完成后总是返回 ERR_MISS_FILE。参考 EXAM13 例子程序。

CH376FileErase(PathName); /* 删除文件，支持多级目录路径 */

删除当前已打开的文件或者指定文件名的文件。如果当前有文件已经打开或者尚未关闭，该子程序直接删除该文件并关闭，如果当前没有文件被打开，那么应该在 PathName 中指定被删除文件的路径名和文件名，格式与 CH376FileOpenPath 相同，不支持通配符。

CH376FileCreatePath(PathName); /* 新建多级目录下的文件并打开，如果文件已经存在则先删除后再新建 */

CH376FileCreate(name); /* 在根目录或者当前目录下新建文件，如果文件已经存在则先删除再新建 */

新建文件，调用该子程序前，应该在 PathName 或者 name 中指定新文件的路径名和文件名，格式与 CH376FileOpenPath 相同，不支持通配符。如果存在同名文件，那么该同名文件将首先被删除，然后再新建文件。如果不希望已有文件被删除，那么应该事先调用 CH376FileOpenPath 确认文件不存在后再新建。新建文件的文件修改日期和时间默认为 2004 年 1 月 1 日 0 时 0 分 0 秒，文件默认长度为 1，如果需要修改则可以调用 CH376DirInfoRead 和 CH376DirInfoSave。示例：

```
strcpy( PathName, "/C51/NEWFILE.TXT" );
/* 新文件名，在 C51 子目录下新建文件 NEWFILE.TXT，要求 C51 已经事先存在 */
CH376FileCreatePath( PathName );  /* 新建文件并打开,如果文件已经存在则先删除再新建 */
```

CH376DirCreatePath(PathName); /* 新建多级目录下的目录(文件夹)并打开，如果目录已经存在那么直接打开 */

CH376DirCreate(name); /* 在根目录下新建目录(文件夹)并打开，如果目录已经存在那么直接打开 */

新建目录（文件夹），用法与新建文件 CH376FileCreatePath 相同。

CH376FileClose(UpdateSz); /* 关闭当前已经打开的文件或者目录(文件夹) */

打开文件或者目录使用完毕后，应该关闭。对于读操作，关闭文件是可选操作。对于写操作，关

闭文件的同时，可以选择让 CH376 芯片自动更新文件长度。UpdateSz 为 TRUE 时自动更新文件长度（如果已经对该文件执行写操作添加了数据），为 FALSE 时禁止自动更新文件长度。在扇区模式下，自动更新的文件长度是以扇区为单位计算的，所以文件长度通常是扇区大小 512 的倍数。在字节模式下，自动更新的文件长度是以字节为单位，所以可以获得适当的长度。在扇区模式下，如果希望文件长度不是扇区大小的倍数，那么单片机可以在关闭文件前调用 CH376WriteVar32 设置 CH376 芯片内部的当前文件长度变量，并且在关闭文件时选择自动更新文件长度。

CH376ByteLocate(offset); /* 以字节为单位移动当前文件指针 */

移动当前已打开文件的指针，用于从指定位置读取数据，或者向指定位置写入数据。例如，单片机希望跳过文件的前 18 字节再读取数据，那么可以在 offset 参数中输入 18，调用该子程序将文件指针移动到 18 个字节后，也就是紧接在后面的读写操作将从第 18 字节开始。对于写操作，如果单片机打算在原文件的尾部继续添加数据，而不希望影响前面的原有数据，那么可以指定一个很大的字节偏移，例如在 offset 参数中输入 0xFFFFFFFF，将文件指针移动原文件的末尾，以便追加数据。该子程序返回后调用 CH376ReadBlock 能够获得当前文件指针所指向的数据在磁盘中的 LBA 扇区号，示例：

```
CH376ByteLocate( 192 );  /* 移动文件指针到第 193 字节，跳过文件头部的 192 个字节 */
/* 以字节为单位进行文件读写操作，读写操作从第 193 字节的位置开始 */
CH376ByteLocate( 0xFFFFFFFF );  /* 移动文件指针到文件末尾,以便在原文件末尾追加数据 */
```

CH376ByteRead(buf, ReqCount, RealCount); /* 以字节为单位从当前文件读取数据块 */

从当前已打开文件中读取数据，每次读取后自动移动文件指针，第二次调用子程序时将从第一次读取数据的后面继续读取数据。输入参数应该在 ReqCount 中指定需要读取的字节数，子程序返回后，读出的数据块被存放在输入参数 buf 指向的缓冲区中。该子程序会自动检查文件是否结束，如果文件已经结束，那么返回时在 RealCount 中为实际读出的字节数，所以判断 RealCount 如果变小就说明文件已经结束。示例：

```
s=CH376ByteRead( MyBuf, 9, &RealCount ); /* 以字节为单位从文件读出数据,准备读 9 字节 */
if ( s!=USB_INT_SUCCESS ) /* 出错 */ else /* 成功 */
if ( RealCount < 9 ) 已经到文件结尾，所以实际读出的长度变小了
/* 在 MyBuf 中为读出的数据块 */
CH376ByteRead( MyBuf, 1234, NULL ); /* 接着刚才的位置向后读数据,准备读 1234 字节 */
```

CH376ByteWrite(buf, ReqCount, RealCount); /* 以字节为单位向当前文件写入数据块 */

向当前已打开文件中写入数据，每次写入后自动移动文件指针，第二次调用子程序从第一次写入数据的后面继续写入数据。输入参数应该在 ReqCount 中指定需要写入的字节数，准备写入的数据块被存放在输入参数 buf 指向的缓冲区中。该子程序会自动检查文件是否结束，并且在需要时会自动分配磁盘空间以便继续写入。该子程序只负责修改或追加数据，而不修改文件长度。如果调用 CH376ByteWrite 写 0 长度的空数据（指定写入字节数 ReqCount 为 0），那么该子程序将更新文件长度。示例：

```
/* 将准备写入的数据块复制到 MyBuf 中 */
s=CH376ByteWrite( MyBuf, 28, NULL ); /* 以字节为单位向文件写入数据,准备写 28 字节 */
if ( s!=USB_INT_SUCCESS ) /* 出错 */ else /* 成功 */
/* 将准备写入的数据块复制到 MyBuf 中 */
CH376ByteWrite( MyBuf, 2345, NULL ); /* 接着刚才的位置向后写数据,准备写 2345 字节 */
```

为了提高读写操作的效率，尽量一次读写较大的数据块，最大为 65535，建议为 512 的倍数。

CH376SecLocate(offset); /* 以扇区为单位移动当前文件指针 */

移动当前已打开文件的指针，用于从指定位置读取数据，或者向指定位置写入数据。例如，单片机希望跳过文件的前 512*2 个字节再读取数据，那么可以在 offset 参数中输入 2，调用该子程序将文件指针移动到 2 个扇区后，也就是从 512*2 个字节处开始。对于写操作，如果单片机打算在原文件的尾部继续添加数据，而不希望影响前面的原有数据，那么可以指定一个很大的扇区偏移，

例如在 offset 参数中输入 0xFFFFFFFF，将文件指针移动原文件的末尾，以便追加数据。该子程序将文件长度取整转换为扇区数后作为最大文件指针，如果文件长度不是扇区大小 512 的倍数，那么文件尾部不足一个扇区的零碎数据部分将被忽略。该子程序返回后调用 CH376ReadBlock 能够获得当前文件指针所指向的数据在磁盘中的 LBA 扇区号，如果是 0xFFFFFFFF 则说明已到文件尾。

CH376SecRead(buf, ReqCount, RealCount); /* 以扇区为单位从当前文件读取数据块 */

从当前已打开文件中读取数据，每次读取后自动移动文件指针，第二次调用时将从第一次读取数据的后面继续读取数据。在调用该子程序前，应该在 buf 中指定缓冲区起始地址，在 ReqCount 中指定准备读取的扇区数。该子程序会自动检查文件是否结束，如果文件已经结束，那么返回时在 RealCount 中为实际读出的扇区数，判断 RealCount 如果变小就说明文件已经结束。如果文件长度不是扇区大小 512 的倍数，那么文件尾部不足一个扇区的零碎数据将被全部读出。不支持 SD 卡。

CH376SecWrite(buf, ReqCount, RealCount); /* 以扇区为单位向当前文件写入数据块 */

向当前已打开文件中写入数据，每次写入后自动移动文件指针，第二次调用时将从第一次写入数据的后面继续写入数据。在调用该子程序前，应该在 buf 中指定缓冲区起始地址，在 ReqCount 中指定准备读取的扇区数。该子程序会检查文件结束簇，并且在需要时会自动分配磁盘空间以便继续写入。该子程序只负责修改或者追加数据，而不修改文件长度。如果调用 CH376SecWrite 写空数据（指定写入扇区数 ReqCount 为 0），那么该子程序将更新文件长度。如果文件长度不是扇区大小 512 的倍数，那么可以在关闭文件前调用 CH376WriteVar32 设置 CH376 芯片内部的当前文件长度变量，并在关闭文件时选择自动更新文件长度。不支持 SD 卡。

CH376DirInfoRead( ); /* 读取当前文件的目录信息 */

查询当前已打开文件的属性、日期、时间、长度等信息。该子程序返回后调用 CH376ReadBlock 能够获得当前文件目录信息结构，参考 FAT_DIR_INFO 结构定义。

在成功调用 CH376DirInfoRead 之后，除非先调用 CH376DirInfoSave 或者 CH376FileClose 或者 CH376EndDirInfo，否则调用 CH376ReadVar32 和 CH376WriteVar32 将无法返回正确数据。

CH376DirInfoSave( ); /* 保存文件的目录信息 */

修改当前已打开文件的属性、日期、时间、长度等的步骤如下：

① 首先，调用 CH376DirInfoRead 读取文件目录信息到 CH376 内存缓冲区；
② 接着，调用 CH376ReadBlock 从 CH376 取得当前文件目录信息；
③ 单片机修改完毕后，调用 CH376WriteOfsBlock 将数据写入 CH376 内存缓冲区；
④ 最后，调用 CH376DirInfoSave 将 CH376 内存缓冲区的数据保存到 U 盘或者 SD 卡中。

CH376DiskCapacity(DiskCap); /* 查询磁盘物理容量,扇区数 */

查询磁盘物理总容量，该子程序返回后 DiskCap 为磁盘物理扇区数。

CH376DiskQuery(DiskFre); /* 查询磁盘剩余空间信息,扇区数 */

查询磁盘剩余空间信息，该子程序在 FAT32 文件系统的磁盘中调用时最快，在 FAT16 文件系统的磁盘中调用时最慢，磁盘容量越大，操作越慢。该子程序返回后 DiskFre 为磁盘剩余扇区数。

CH376GetLongName(PathName, LongName); /* 由短文件名或目录名获得相应的长文件名 */

由短文件名或者目录(文件夹)名获得相应的长文件名。需要在 PathName 中输入短文件名的完整路径，需要 LongName 指向缓冲区以接收长文件名，返回的长文件名是以 UNICODE 小端编码，以双 0 表示结束。该子程序使用了全局缓冲区 GlobalBuf 的前 34 个字节。

CH376CreateLongName(PathName, LongName); /* 新建具有长文件名的文件,关闭文件后返回 */

新建具有长文件名的文件，关闭文件后返回。需要在 PathName 中输入短文件名的完整路径，其中短文件名必须事先参考 FAT 规范由长文件名自行产生，需要在 LongName 中输入以 UNICODE 小端编

码的以双 0 结束的长文件名。该子程序使用了全局缓冲区 GlobalBuf 的前 64 个字节。

5.2. 硬件 I/O 接口子程序

硬件 I/O 接口子程序用于单片机与 CH376 芯片之间交换数据。应用程序应该根据实际的硬件连接方式（并口或者 SPI 或者异步串口）实现 I/O 接口子程序。

```
void    CH376_PORT_INIT( void );        /* CH376 通讯接口初始化 */

void    xEndCH376Cmd( void );           /* 结束 CH376 命令,仅用于 SPI 接口方式 */

void    xWriteCH376Cmd( UINT8 mCmd );   /* 向 CH376 写命令 */

void    xWriteCH376Data( UINT8 mData ); /* 向 CH376 写数据 */

UINT8   xReadCH376Data( void );         /* 从 CH376 读数据 */

UINT8   Query376Interrupt( void );      /* 查询 CH376 中断(INT#引脚为低电平) */
```

6、应用程序示例

针对一些常见的实际应用提供程序范例，分别位于 EXAM??各个子目录中。目前为 MCS51 单片机提供了多个示例程序，其它单片机也可以参考。不同单片机的 C 语言示例程序基本通用，只需要修改硬件 I/O 等相关部分，main 主程序通常无需修改，重新编译和链接就可以使用。

6.1. 单片机控制 USB 主从模式切换，示例 EXAM0

C 语言示例程序，演示 USB-HOST 主机接口和 USB-DEVICE 设备接口的应用。

默认工作于 USB-HOST 主机方式，当有 USB 设备连接时自动处理，需要作为 USB 设备与计算机通讯时，可以按评估板上的按钮由主程序进行切换。

主机方式下，插入 U 盘后，该程序将 U 盘中的/C51/CH376HFT.C 文件中的前 200 个字符显示出来，如果找不到文件，那么该程序将显示当前目录下所有文件名。

设备方式下，CH376 的 INT#引脚采用中断方式，程序结构为“请求+应答模式”，强调可靠性和交互性，不追求传输速度。计算机端可以通过 CH372/CH375/CH376 的调试工具中的 MCS51 监控工具程序 CH37XDBG.EXE 实现对 MCS51 单片机的“完全”控制，可以读写 MCS51 单片机的任意外部 RAM、内部 RAM 以及绝大多数 SFR，当然也能够进行数据通讯。

涉及到 USB 主从模式切换，字节模式的文件读写，文件打开/关闭/新建，USB 设备方式下的数据通讯等。

6.2. 文件复制和文件枚举，示例 EXAM1

C 语言示例程序，该程序演示字节读写，文件枚举，简单的文件复制。用于将 U 盘或者 SD 卡中的/C51/CH376HFT.C 文件中的小写字母转成大写字母后，写到新建的文件 NEWFILE.TXT 中。如果找不到原文件 CH376HFT.C，那么该程序将显示 C51 子目录下所有以 CH376 开头的文件名，并新建 NEWFILE.TXT 文件并写入提示信息。如果找不到 C51 子目录，那么该程序将显示根目录下的所有文件名，并新建 NEWFILE.TXT 文件并写入提示信息。

涉及到字节模式文件读写，文件打开/关闭/新建，文件枚举/搜索等。

6.3. 汇编 ASM 语言下的文件读写，示例 EXAM5

ASM 语言示例程序，该程序演示字节读写，用于将 ADC 模数采集的数据添加到 U 盘或者 SD 卡文件 MY_ADC.TXT 中。如果文件存在那么将数据添加到文件末尾，如果文件不存在那么新建文件后添加数据。

涉及字节模式文件读写，文件打开/关闭/新建，移动文件指针等。

6.4. 字节模式文件读写，示例 EXAM7

C 语言示例程序，该程序用于将 ADC 模数采集的数据添加到 U 盘或者 SD 卡文件 MY_ADC.TXT 中，如果文件存在那么将数据添加到文件末尾，如果文件不存在那么新建文件后添加数据。

涉及到字节模式的文件读写，文件打开/关闭/新建，移动文件指针/添加数据等。

6.5. 扇区模式文件读写，示例 EXAM8

C 语言示例程序，该程序演示扇区读写，修改文件长度，查询磁盘剩余空间，用于将 ADC 模数采集的数据添加到 U 盘文件 MY_ADC.TXT 中，为提高速度而以扇区为单位读写 U 盘文件，不支持 SD 卡。该程序演示在扇区模式下处理零碎数据，同时兼顾操作方便和较高速度，需要单片机系统提供足够的 RAM 作为文件数据缓冲区 FILE_DATA_BUF，该程序使用了 16K 外部 RAM。

涉及到扇区模式的文件读写，文件打开/关闭/新建，扇区模式的移动文件指针/添加数据，以及非扇区大小 512 倍数的任意文件长度的处理等。

6.6. 创建子目录的应用，示例 EXAM9

C 语言示例程序，该程序用于演示创建子目录，然后在该子目录下再创建二级子目录，以及打开多级目录下的文件或者目录。由于 FAT12 或者 FAT16 的文件系统中，根目录下不能超过 512 个文件或者目录，所以在需要大量产生新文件或者需要按文件类型进行分类处理时，可以参考该程序。

涉及到字节模式的文件（目录）写，文件（目录）打开/关闭/新建等。

6.7. 处理文件目录项的应用，示例 EXAM10

C 语言示例程序，该程序用于演示处理文件目录项 FAT_DIR_INFO，例如：修改文件名、设置文件的创建日期和时间等。

涉及到文件目录项的结构、查询和修改文件信息等。

6.8. 处理小写文件名和长文件名的应用，示例 EXAM11

C 语言示例程序，该程序是长文件名操作示例，包括创建长文件名，和从短文件名获得对应的长文件名，仅供具有一定技术水平和相关知识的高级用户参考。

涉及到新建/打开/关闭目录，字节模式的目录读写，文件目录项的结构，长文件名结构等。

6.9. 检查 U 盘写保护和安全移除 U 盘，示例 EXAM12

C 语言示例程序，该程序用于演示检查 U 盘是否写保护，演示模拟计算机端的安全移除，也可以参考用于自行处理其它命令。

涉及到 USB 配置、USB MassStorage 和 SCSI 规范，新建文件/关闭，字节模式写文件等。

6.10. 搜索和枚举文件名的应用，示例 EXAM13

C 语言示例程序，该程序用于演示搜索和枚举文件名，列出指定目录下的所有文件，为整个 U 盘的文件建立索引表，优化某些慢启动 U 盘执行 CH376DiskMount 时的等待时间等，仅供具有一定技术水平和相关知识的高级用户参考。

涉及到打开/关闭目录，字节模式的目录读操作，文件目录项的结构等。

7、其它说明和建议

7.1. 硬件方面

⑴ 如果用频率计测量 CH376 的振荡频率要考虑探头电容对频率的影响，应该用 10:1 高频探头，普通晶体基本上可以满足晶体 X1 的 0.4‰精度要求。简单应用中也可以使用陶瓷晶体降低成本。

⑵ 为了降低电磁辐射，并减少来自外界的干扰，晶体 X1 的金属外壳应该接地，晶体 X1 应该尽量靠近 CH376，相关的 PCB 走线应该尽量短，并且可以在周边环绕接地线或者敷铜。时钟信号线的 PCB 周边不应该有大电流布线或者强脉冲信号布线，以避免引入干扰。

⑶ USB 数据线 D+和 D-应该平行布线，长度保持差不多，并且应该尽量减少信号线上的过孔和焊盘以及分叉，两侧可以环绕接地线或者敷铜，USB 信号线的 PCB 周边不应该有大电流布线或者强脉冲信号布线，以避免引入干扰。更详细的说明可以参考 USB 规范。

⑷ USB 的 GND 与 CH376 的 GND 以及公共地线应该接触可靠，PCB 的 GND 走线不宜太长，减少由于较大电源电流流过 GND 线而在两端之间产生的电压差。

⑸ 如果不直接连接 USB 插座，而是通过排线等引到主机板或者其它位置，那么应该像上述 PCB 布线一样，D+和 D-紧靠，两侧各安排一根 GND 线。如果距离较长，那么应该使用标准 USB 信号线。

⑹ 与 USB 设备或者计算机 USB 端口相连的 USB 线应该符合 USB 规范，对于全速 12Mbps 信号，USB 线应该是带屏蔽层的绞线，线的一端 USB 插头（或者插座）的外壳与另一端的 USB 插头的外壳相通，但是与 USB 信号线中的 GND 线不通，也就是说，屏蔽层应该独立于 4 根 USB 信号线。

⑺ 对于需要频繁带电插拔 USB 设备的应用以及静电较强的环境，建议在电路中增加 USB 信号瞬变电压抑制器件或者 ESD 保护器件（例如 CH412 等），为 CH376 的 USB 引脚 D+和 D-提供进一步的保护。详细说明可以参考 CH375 和 CH376 电路设计注意事项 README.PDF 文档。

⑻ 如果 SD 卡需要频繁带电插拔，建议在电路中为 CH376 与 SD 卡座连接的引脚增加 ESD 保护器件。

⑼ USB 电源必须是 5V，对于 USB-HOST 应用，必须对外部的 USB 设备例如 U 盘提供 5V 电源，供电电流视 U 盘而定，考虑 U 盘峰值电流，通常供电能力不能少于 200mA，建议为 500mA 以上。

⑽ 如果操作 USB 外置硬盘或者耗电较大的 USB 闪存盘，需要考虑其电源供应，确保提供足够的工作电流，否则在其插入过程以及读写过程中会导致电源电压波动，甚至导致 CH376 以及单片机复位。建议在电源与地之间并联较大的电解电容，或者为 USB 插座单独提供一组 5V 电源，或者用直流电阻较小的电感代替电阻以减少对 CH376 的影响。

⑾ CH376 芯片本身的供电电压可以选择 5V 电源或者 3.3V 电源，在 3.3V 电源时须将 V3 引脚与 VCC 引脚短接后共同输入 3.3V，在 5V 电源时须在 V3 引脚外接一个 4700pF 或者 0.01uF 的电容。

⑿ 如果 CH376 工作于 5V 电源电压并且需要连接 SD 卡（SD 卡是 3.3V 电源电压），那么有两种方法：一是标准方法，在 CH376 和 SD 卡之间加电平转换芯片，二是简易方法，在所有信号线中串联 1K 左右的电阻。

⒀ 如果需要减小电流消耗，可以在空闲时使 CH376 进入低功耗睡眠挂起状态，当有 USB 设备插拔时 CH376 会自动唤醒。在 CH376 睡眠期间，应该使 CH376 的各个 I/O 引脚（除 RSTI 引脚）处于悬空或者高电平状态，避免产生不必要的上拉电流。

7.2. 单片机程序方面

⑴ 有些 USB 设备包括 U 盘，在刚插入 USB 插座后，不能立即进入工作状态，所以主程序可以在检测到 USB 设备连接后，等待数百毫秒再对其进行操作。详细做法可以参考 EXAM1 和 EXAM13 等。

⑵ 单片机的 C 语言效率比汇编语言低，以 MCS51 单片机为例，纯 C 语言数据读写的速度可能只有汇编语言速度的一半，所以，根据需要部分子程序可以嵌入汇编。如果是频率为 24MHz 的标准 MCS51 单片机传输文件数据（附加说明：MCS-51 单片机每复制一个字节至少需要 8 个机器周期即 4uS），那么传输速度约为每秒 50KB 到 200KB，文件越零碎，传输速度越慢。

⑶ 做好 U 盘操作出错后的分析处理和状态恢复。例如，分析返回错误码，如果是 U 盘断开（意外拔出）则重新等待 U 盘插上，其它错误则可先调用 CH376DiskMount 检查 U 盘是否就绪（对于 SD 卡

则尝试获得磁盘物理总容量 CH376DiskCapacity)，然后再重新打开文件或者新建文件并读写等.如果仍然出错，则应该调用 CH376WriteVar8(VAR_DISK_STATUS, DEF_DISK_DISCONN)强行清除 CH376 芯片内部的磁盘及文件状态，然后再调用 CH376DiskMount 重新初始化 U 盘和 SD 卡。

(4) 以字节为单位的文件读写子程序，占用 RAM 相对较少，能够自动处理文件长度，使用较为方便。建议尽量缓冲和集中多个零碎数据，然后合并起来成批成块写入，减少擦写次数。建议单次写入字节数尽量是扇区大小 512 的倍数，避免频繁地向 U 盘中的文件写入零碎的数据，那样会缩短 U 盘或者 SD 卡中闪存的使用寿命（因为闪存只能进行有限次擦写）。例如，某 U 盘闪存 FLASH 的擦除次数是 100 万次，如果两秒钟写一次，FLASH 盘就要至少擦除一次，计算一下这个 U 盘只能用 23 天。实际上大多数 U 盘所用的 NAND 闪存芯片通常只有 10 万次擦写寿命。

(5) 在单片机向 U 盘写数据的时候禁止将 U 盘拔出，否则会导致某些 U 盘损坏，实际上是 FLASH 数据变成无效，U 盘厂家通常可以修复。有些 U 盘将产品信息例如名称、型号甚至控制程序（节约成本）放在闪存 Flash 的特定区域中，偏偏对这块闪存区域又没有保护措施，一旦应用程序因意外原因误写该闪存区域，将导致该 U 盘无法正常使用，有的现象是 U 盘名称、型号竟然变了。

(6) 有些 U 盘在数据刚刚写入后还有一个延后写 FLASH 的过程，所以这时也不允许拔出，这种盘一般会在刚写进数据后还能看到指示灯在闪烁，直到 U 盘内部完全写入完成。建议在向 U 盘写入数据后稍做延时再允许将盘拔出，个别 U 盘甚至要延时 3 秒才能真正写完。建议参考 EXAM12 实现 U 盘安全移除（理论上可行）。

(7) U 盘和 SD 卡的物理存储介质是闪存 FLASH，理论上有寿命，会有永久失效的可能性。建议写完 U 盘后仍然保存本机的数据备份，直到确信数据可以删除后再清理备份。或者再将数据读出进行校验确信写进 U 盘的数据是正确的。在计算机上进行测试时，曾经发现个别 U 盘在闪存局部失效时实际写入 U 盘的数据出错而 U 盘本身不提示出错的现象，结果使计算机误以为写入正确。

(8) 在 WINDOWS 2000 或者 XP 下的控制面板/管理工具/计算机管理中有磁盘管理工具，可以将 U 盘格式化成指定的 FAT12、FAT16 或者 FAT32 文件系统，当总容量除以分配单元大小后的结果小于 4085 时是 FAT12，大于 65525 时是 FAT32，否则是 FAT16。对于已经格式化过的 U 盘，可以使用命令行工具 CHKDSK 分析，点击[开始]选择[运行]，输入“CHKDSK 盘符:”分析指定盘符的磁盘，显示分配单元的大小和总数。分配单元较大时，通常读写效率稍高，分配单元较小时，通常会节约磁盘容量。建议：如果 U 盘容量小于 16MB，可以使用 FAT12 文件系统格式；如果 U 盘容量小于 512MB，优先使用 FAT16 文件系统格式；如果 U 盘容量大于 512MB，可以使用 FAT32 文件系统格式。

(9) 下表是单片机通过 CH376 读写 U 盘文件时的速度（单位为字节/每秒），是读取 5MB 大文件时测得的。如果每次读取的数据块较小，那么读速度会略有下降。而写速度与 U 盘本身的技术方案有关，在大数据块写入时的速度通常是读速度的 70%到 90%，在小数据块写入时速度明显下降。对于普通的 MCS51 单片机，从外部 RAM 中读出数据并写到 CH376 或者从 CH376 读出数据写到外部 RAM 中，汇编指令每复制一个字节至少需要 8 个机器周期即 4uS，如果是 C 语言程序还要慢一倍。

单片机硬件类型：主频	I/O 通讯 接口方式	扇区读 每次 16K	扇区写 每次 16K	字节读 每次 512	字节写 每次 512
12 时钟 MCS51 单片机：24MHz 括号中汇编语言优化 RAM 读写	并口总线	60K (200K)	55K (110K)	40K (90K)	20K (50K)
12 时钟 MCS51 单片机：40MHz 括号中汇编语言优化 RAM 读写	并口总线	100K (280K)	90K (145K)	65K (135K)	32K (63K)
12 时钟 MCS51 单片机：18MHz	异步串口 115200bps	10.5K	9K	10K	8K
AVR 单片机：16MHz	并口总线	455K	185K	385K	87K
ARM 单片机：45MHz	并口总线	500K	210K	360K	96K
ARM 单片机：45MHz	硬件 SPI 20MHz	456K	185K	358K	90K
大多数 16 位或 32 位单片机		200K-500K	100K-300K	100K-400K	50K-100K