USB 芯片 CH375 的评估板说明及应用参考

版本： 3

http://wch.cn

1、概述

本评估板采用 USB 总线接口芯片 CH375 和通用的 MCS51 系列单片机构成（后面的大多数说明也适用于其它单片机），用于演示 CH375 或者 CH374 的 USB-HOST 主机接口功能和 USB 设备接口功能。本评估板可以读写 U 盘（闪盘、USB 闪存盘、USB 外置硬盘、USB 读卡器等）；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 CH372 评估板或者 CH375 评估板进行对连，互传数据。

另外，评估板通过 USB 总线转接芯片 CH341 提供 USB 与串口的转换，用于演示 CH341 通过 USB 提供仿真串口的应用，可以用于将原串口产品升级到 USB 产品。

2、评估板的硬件

评估板的原理图和 PCB 请参考 CH375SCH.PDF 文档。下面是元器件说明。

评估板中的主要器件有单片机 U1，USB 主机和设备双用接口芯片 CH375，USB 转串口芯片 CH341。其中，CH375 和 CH341 同时提供 SOP28 封装的贴片焊接位置和 DIP28 封装的插座，初期调试时建议用 SOP28 转 DIP28 的转换板。为了避免引线太长引入信号干扰，使用转换板时，建议用转换板上自带的时钟振荡电路（晶体和电容）。CH375 的位置也可以用于 CH374 芯片。

评估板中 U9 是 SRAM-62256，提供了 32KB 的外部 RAM，地址是 0000H 到 7FFFH，用于在单片机读写 U 盘时进行数据缓冲。实际应用中可以不需要这么大的缓冲区，有些示例程序只需要使用 512 字节或者 2K 字节的外部 RAM，所以最终产品也可以用内置 1KB 或者 2.2K 外部 RAM 的 51 单片机代替评估板中的单片机及 U9，从而节约产品体积和综合成本。

评估板有两个 USB 端口：左边的 P1 连接 CH375，如果演示 USB 主机方式的应用，那么 P1 应该连接外部的目标 USB 设备，例如 U 盘等，如果演示 USB 设备方式的应用，那么 P1 应该连接计算机；右边的 P2 连接 CH341，用于连接计算机提供 USB 仿真串口。

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 晶体完全可以满足要求，另外，振荡电容 C1、C2 的容量也会少量影响振荡频率。强烈建议缩短相关引线的长度，以减少干扰。

电阻 R1 的阻值为 1 欧姆到 10 欧姆之间，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 CH375 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 C24 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。

电容 C3 和 C5 用于向 CH375 和 CH341 提供可靠的上电复位，是可选器件，容量为 0.01uF 到 1uF。电容 C4 和 C6 用于内部电源节点退耦从而降低 USB 传输过程中的 EMI，是可选器件，容量为 1000pF 到 0.1uF，可以选用普通的 103 贴片电容。如果实际产品电路中有μP 监控电路，那么应该为 CH375 和单片机提供同一个复位信号，并且去掉 C3 或者 C5。

端口 P5 为外部电源输入插座，输入电压为 8V 到 15V 之间，极性为外负内正，建议外部输入电流不要小于 200mA，否则在 U 盘读写期间可能电流不足。

端口 P3 包含电源引脚和单片机的串口引脚，可以用于连接外部 5V 电源，或者提供单片机串口与外界其它电路的连接。如果不从端口 P5 输入外部电源，也可以从 P3 端口的 1 脚和 2 脚直接输入稳压后的 5V 电源。

端口 J2 是 RS232 串口，可以通过 3 芯串口交叉线连接计算机或者连接其它 RS232 设备，串口交叉线要求两头 9 孔，2 脚与 3 脚交叉，5 脚直通。

端口 P4 是 CH341 的外界接口，提供 TTL 电平的异步串口，可以连接外界单片机的串口引脚或者连接 TTL 电平的串口设备，用于将原串口产品升级为 USB 产品（实际是由 CH341 提供 USB 仿真串口）。如果与 CH341 芯片相连接的串口产品或者单片机不需要硬件速率控制（未使用 CH341 的发送允许输入

TEN#和接收就绪输出 RDY#），那么应该将 P4 的 5 脚 RDY 和 6 脚 TEN 短接，使 CH341 的 TEN#引脚为低电平，直接允许 CH341 串口发送。该端口还可用于对支持串口下载的单片机进行程序下载。

端口 P6 为预留的对外接口，可以用于连接 CH374 评估板或者其它外部电路。

当前版本的评估板已经去掉 U11，在以前版本评估板上的 EEPROM 器件 U11 用于对 CH341 进行配置，例如指定 USB 的 VID 和 PID 以及工作模式等，通常不需要用到。另外，U11 也可以用于演示 CH341 转 SCL/SDA 两线串行接口的应用，只要连接 CH341 和 PC 机，应用程序就可以通过 CH341 读写 U11 中的数据。如果使用 CH341 并口驱动程序，那么应用程序通过 API 调用读写 U11 中的数据；如果使用 CH341 串口驱动程序，那么应用程序可以在 300bps 波特率下以专用命令包读写 U11 中的数据。

在评估板示例程序中，可能会用单片机 U1 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 J2 连接到计算机使用串口监控/调试工具软件查看。U7（MAX232 或者 ICL232 等）用于提供串口连接的 RS232 电平转换，如果计算机没有串口，或者串口已经被其它设备占用，那么 CH341 可以通过与计算机连接的 USB 线提供仿真串口。

下表是 6 脚跳线 JP5 的有关设置，用于选择单片机串口、CH341 串口、RS232 串口的组合功能，表中左边 JP5 跳线设置的黑色块代表短路子连接，数字代表引脚号，表中所示跳线位置方向与评估板 PCB 中的实际方向一致。

JP5 跳线设置	JP5 跳线说明	实际物理连接及用途
2 4 6 1 3 5	4 脚连 2 脚 3 脚连 5 脚	单片机串口通过 232 芯片连接到 RS232 端口 J2， 再由 J2 通过交叉线与 PC 计算机的普通串口相连接， 可以用 PC 计算机普通串口监控单片机的串口输出
2 4 6 1 3 5	4 脚连 6 脚 3 脚连 1 脚	单片机串口通过 CH341 芯片连接到 USB 端口 P2， 再由 P2 通过 USB 线与 PC 计算机的 USB 端口相连接， 可以用 PC 计算机 CH341 仿真串口监控单片机串口输出
2 4 6 1 3 5	1 脚连 2 脚 5 脚连 6 脚 未连单片机	CH341 串口通过 232 芯片连接到 RS232 端口 J2， P2 连接 PC 计算机 USB 端口，J2 连接 RS232 串口设备， 可以将原 RS232 串口产品升级为 USB 产品（仿真）
2 4 6 1 3 5	全部断开 未连单片机	CH341 串口直接连接到外界端口 P4（USB 转串口）， P2 连接 PC 计算机 USB 端口，P4 连接 TTL 电平串口设备， 可以将原 TTL 串口产品升级为 USB 产品（仿真）

上表中的最后一项是从 P4 端口提供 TTL 电平的 USB 仿真串口，如果 P4 所连接的串口产品不需要硬件速率控制，那么应该将 P4 的 5 脚 RDY 和 6 脚 TEN 短接。

评估板上有多个 LED 指示灯和多个跳线，其中，L6 为电源指示灯，L1 为 CH341 连接计算机初始化成功的指示灯和 USB 设备连接的指示灯，L7 为工作于 USB 设备方式的 CH375 连接计算机初始化成功的指示灯，L2-L5 由应用程序自行定义，跳线 JP1-JP4 及按键 K1 由应用程序自行定义。

评估板可以用于演示 U 盘文件读写，随板提供的大多数示例程序都用到 LED，其中，L2 为 U 盘连接指示灯，L3 为 U 盘正在操作指示灯，L4 为 U 盘正在写数据指示灯。

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。

JP9 用于选择由 CH375 给单片机 U1 复位还是由阻容 R6、C28 或者按键 K2 给单片机 U1 复位。JP10 用于选择由 C3 给 CH375 复位还是由单片机的 I/O 引脚给 CH375 复位。

JP6、JP7 预留给 CH374 芯片（与 CH375 引脚类似，DIP28 转换板可以直接互换），JP6 对应 UEN 引脚，默认是 1-2 脚短接，JP7 默认是断开，短接后用于 CH374 向单片机提供可编程时钟。

JP8 预留给支持 CH375/CH372/CH374 的 MCS51 单片机通过 USB 进行程序下载，JP8 默认是断开的，先短接 JP8 再通过 P1 端口的 USB 连接计算机，3 秒内再断开 JP8，即可支持 USB 下载。

3、示例程序

关于 CH341 仿真串口的说明请参考 CH341 的相关资料。

产品光盘或者网站上提供了 CH375 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 CH375 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。

如果在评估板中做测试，那么 CH375 的端口地址是（也可以用其它地址）：

```
unsigned char volatile xdata CH375_CMD_PORT _at_ 0xBDF1; /* CH375 命令端口的 I/O 地址 */
unsigned char volatile xdata CH375_DAT_PORT _at_ 0xBCF0; /* CH375 数据端口的 I/O 地址 */
#define CH375_INT_WIRE  INT0        /* 连接 CH375 的 INT#引脚，用于查询中断状态 */
    /* 对于 CH375B 芯片，如果不定义 CH375_INIT_WIRE，那么可以查询命令口的位 7 代替 */
```

如果用于非 MCS51 单片机，需要修改以下与单片机硬件有关的子程序：

```
/* 延时 2 微秒，需要根据硬件实际情况做调整，不精确 */
void    delay2us( void ) { unsigned char i; for ( i = 2; i != 0; i -- ); }

/* 延时 1 微秒，需要根据硬件实际情况做调整，对于大多数 MCS51 单片机应该省去该子程序 */
void    delay1us( void ) { unsigned char i; for ( i = 1; i != 0; i -- ); }

void CH375_WR_CMD_PORT( unsigned char cmd ) {
/* 向 CH375 的命令端口写入命令，周期不小于 4uS（CH375B 为 3uS），如果单片机较快则延时 */
    delay2us();
    CH375_CMD_PORT=cmd;
    delay2us();
}

void CH375_WR_DAT_PORT( unsigned char dat ) {
/* 向 CH375 的数据端口写入数据，周期不小于 1uS（CH375B 为 0.6uS），如果单片机较快则延时 */
    CH375_DAT_PORT=dat;
    delay1us();  /* 因为大多数的 MCS51 单片机较慢，所以实际上无需延时 */
}

unsigned char CH375_RD_DAT_PORT( void ) {
/* 从 CH375 的数据端口读出数据，周期不小于 1uS（CH375B 为 0.6uS），如果单片机较快则延时 */
    delay1us();  /* 因为大多数的 MCS51 单片机较慢，所以实际上无需延时 */
    return( CH375_DAT_PORT );
}
```

(1) 进入 USB 主机模式

CH375 支持 USB 设备方式和 USB 主机方式，要想读写 U 盘或者操作 USB 打印机，必须先设置 CH375，使其工作于 USB 主机方式。以下是例子，节选自 CH375EVO.C

```
unsigned char   mCH375Init( ) {  /* 设置 CH375 为 USB 主机方式 */
    unsigned char   i;
    CH375_WR_CMD_PORT( CMD_SET_USB_MODE );  /* 设置 USB 工作模式 */
    CH375_WR_DAT_PORT( 6 );  /* 模式代码，主机方式，自动检测 USB 设备连接 */
    for ( i = 0xff; i != 0; i -- ) {  /* 等待操作成功，通常需要等待 10uS-20uS */
        if ( CH375_RD_DAT_PORT( ) == CMD_RET_ABORT ) break;  /* 操作失败 */
    }
    if ( i == 0 ) return( 0 );  /* 操作成功 */
    else return( 0xff );  /* CH375 出错，例如芯片型号错或者处于串口方式或者不支持 */
}
```

(2) 将 U 盘当作存储器进行读写

这种方式下，仅仅是将 U 盘当作可移动的存储器（闪存或者硬盘），所以读写方法与读写闪存差不多，操作简单，速度快，只要几十条语句就可以读写数据。

由于计算机要求 U 盘中的数据按照特定的文件格式进行组织，所以如果直接将 U 盘当作存储器，而不符合文件格式，那么写入数据后的 U 盘插到计算机中，只能通过工具程序取出数据，或者可以参考 CH375UD.C 编写计算机程序以特殊方法读写无文件格式 U 盘中的数据。

作为存储器，U 盘具有一维的线性地址 LBA，基本的存储单元是扇区（通常每扇区是 512 字节，部分 U 盘的扇区是 2K 字节或者其它值），16 兆容量的 U 盘通常有 32768 个扇区，LBA 地址是从 0 到 32767，单片机可以任意指定 LBA 地址，然后以扇区为单位进行数据读写。

为了提高写操作的效率，有必要了解 U 盘中所用闪存的一些特性，因为某些闪存必须进行块擦除，所以对于写操作，最佳的方法是一次写入 32 个扇区（某些闪存要求 64 个扇区或者更多），并且线性起始扇区号 LBA 应该是 32 的倍数，这样写操作的速度将是最高的，否则可能写操作的速度会下降一半以上。读操作没有这类要求。

下面是示例子程序，节选自 CH375EVO.C

```
/* 从 U 盘读取多个扇区的数据块到缓冲区，可以读写 U 盘中的任意位置的数据 */
unsigned char   mReadSector( unsigned long iLbaStart, unsigned char iSectorCount ) {
/* iLbaStart 是准备读取的线性起始扇区号 LBA, iSectorCount 是准备读取的扇区数 */
    unsigned char mIntStatus;
    unsigned char *mBufferPoint;
    unsigned int  mBlockCount;
    unsigned char mLength;
    CH375_WR_CMD_PORT( CMD_DISK_READ );  /* 从 USB 存储器读数据块 */
    CH375_WR_DAT_PORT( (unsigned char)iLbaStart );  /* LBA 的最低 8 位 */
    CH375_WR_DAT_PORT( (unsigned char)( iLbaStart >> 8 ) );
    CH375_WR_DAT_PORT( (unsigned char)( iLbaStart >> 16 ) );
    CH375_WR_DAT_PORT( (unsigned char)( iLbaStart >> 24 ) );  /* LBA 的最高 8 位 */
    CH375_WR_DAT_PORT( iSectorCount );  /* 扇区数 */
    mBufferPoint = &DATA_BUFFER;  /* 指向缓冲区起始地址 */
    for ( mBlockCount=iSectorCount*SectorSize/64; mBlockCount!=0; mBlockCount-- ){
        /* 数据块计数,SectorSize 由 CMD_DISK_SIZE 获得,大多数 U 盘的 SectorSize 是 512 */
        mIntStatus = mWaitInterrupt( );  /* 等待中断并获取状态 */
        if ( mIntStatus == USB_INT_DISK_READ ) {  /* USB 存储器读数据块,请求数据读出 */
            CH375_WR_CMD_PORT( CMD_RD_USB_DATA );  /* 从 CH375 缓冲区读取数据块 */
            mLength = CH375_RD_DAT_PORT( );  /* 后续数据的长度 */
            while ( mLength -- ) {  /* 根据长度读取数据 */
                *mBufferPoint = CH375_RD_DAT_PORT( );  /* 读出数据并保存 */
                mBufferPoint ++;
            }
            CH375_WR_CMD_PORT( CMD_DISK_RD_GO );  /* 继续执行 USB 存储器的读操作 */
        }
        else break;  /* 返回错误状态 */
    }
    if ( mBlockCount == 0 ) {  /* 数据传输完成 */
        mIntStatus = mWaitInterrupt( );  /* 等待中断并获取状态 */
        if ( mIntStatus == USB_INT_SUCCESS ) return( 0 );  /* 操作成功 */
    }
    return( mIntStatus );  /* 操作失败 */
}
```

主程序节选

```
    mInitDisk( );  /* 初始化 U 盘，实际是识别 U 盘的类型，不影响 U 盘中的数据 */
/* 每次 U 盘拔出再重新插上之后，在所有读写操作之前必须对 U 盘进行初始化 */
    mReadSector( 0, 1 );  /* 从 0#扇区开始读 1 个扇区的数据 */
    mWriteSector( 0x1234, 32 ); /* 从 1234H#扇区开始写 32 个扇区的数据 */
```

(3) 按照文件格式读写 U 盘

这种方式下，单片机需要分析 U 盘的已有文件系统，并按照其格式进行数据读写，操作复杂，速度也有所下降，很多操作需要数百条语句才能完成。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。

为了兼容 WINDOWS 下的文件系统，小容量 USB 闪存盘常用的文件系统是 FAT12 或者 FAT16，超过 128M 容量的 U 盘或者 USB 外置硬盘，常用的文件系统是 FAT32。

由于文件系统对于单片机而言，处理起来比较复杂，所以我们通过子程序库提供 U 盘文件操作的 API。有关这些 API 的进一步说明可以参考 CH375HF.PDF 文档。

以下是示例程序，节选自 CH375HFT.C

```
    strcpy( mCmdParam.Open.mPathName, "\\C51\\CH375HFT.C" );
  /* 要操作的文件名称是 CH375HFT.C，该文件在 C51 子目录下，C 语言的双斜杠实际是单斜杠 */
    CH375FileOpen( );  /* 打开文件,实际应用还需判断该操作是否成功 */
    mCmdParam.ReadX.mSectorCount = 4;  /* 读取 4 个扇区（4*SectorSize 数据） */
    mCmdParam.ReadX.mDataBuffer = 0x0200;  /* 将读出数据放到 0200H 开始的缓冲区中 */
    CH375FileReadX( );  /* 以扇区为单位从文件读取数据，如果文件比较大，可以多次读取 */
    CH375FileClose( );  /* 关闭文件 */
/* 数据处理，示例程序是将原文件中的小写字符转换为大写字符，然后保存到一个新文件中 */
    strcpy( mCmdParam.Create.mPathName, "/NEWFILE.TXT" );
/* 新建一个文件，名称是 NEWFILE.TXT，在根目录下，下面以字节为单位读写该文件 */
    CH375FileCreate( );  /* 新建文件并打开，如果文件已经存在则先删除后再新建 */
    strcpy( mCmdParam.ByteWrite.mByteBuffer, "准备写入新文件中的字符串" ); /* 数据 */
    mCmdParam.ByteWrite.mByteCount=strlen(mCmdParam.ByteWrite.mByteBuffer);  /* 长度 */
    CH375ByteWrite( );  /* 以字节为单位向文件写入数据块，单次长度不超过 MAX_BYTE_IO */
    unsigned short  Dat = 12345;  /* 下面将变量 Dat 格式化为字符串输出到文本文件中 */
    mCmdParam.ByteWrite.mByteCount = /* 转换后的字符串长度由下面的 sprintf 返回 */
        sprintf(mCmdParam.ByteWrite.mByteBuffer, "变量 Dat=%d", Dat);  /* 输出文本 */
    CH375ByteWrite( );  /* 以字节为单位向文件写入数据块，单次长度不超过 MAX_BYTE_IO */
    CH375FileClose( );  /* 字节模式下关闭文件会自动计算文件长度 */
```

(4) 操作其它 USB 设备（更多详细说明可以参考 CH375HST 主机方式应用参考）

被操作的 USB 设备应该支持 USB1.1 或者 USB2.0 的 12Mbps 速度，如果打算操作低速 1.5Mbps 的 USB 设备，那么应该使用 CH375B 芯片或者 CH374 芯片。

CH375 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。

以下是示例，节选自中断方式 CH375EV1.C（查询方式示例是 CH375EV2.C）

```
void    mCtrlGetDescr( unsigned char type ) { /* 执行控制传输: 获取 USB 描述符 */
    mIntStatus = 0;  /* 清中断状态 */
    CH375_WR_CMD_PORT( CMD_GET_DESCR );  /* 控制传输-获取描述符 */
    CH375_WR_DAT_PORT( type );  /* 0:设备描述符, 1:配置描述符 */
    while ( mIntStatus == 0 );  /* 等待操作完成，mIntStatus 由中断服务程序更新 */
}
```

主程序节选

```
    while ( mDeviceOnline == 0 );  /* 等待 USB 设备连接，连接后应该适当延时几毫秒 */
    mResetUsbDevice( );  /* 复位 USB 设备 */
    mCtrlGetDescr( 1 );  /* 获取设备描述符 */
    len = mReadCH375Data( DATA_BUFFER );  /* 读取设备描述符数据 */
    printf( "Device descr data len: %d, data: ", len );  /* 显示描述符数据 */
    for ( i=0; i<len; i++ ) printf( "%02X,", (unsigned int)DATA_BUFFER[i] );
    mCtrlSetAddress( 5 );  /* 设置 USB 地址,地址值为 1 到 7EH */
    mCtrlGetDescr( 2 );  /* 获取配置描述符 */
    len = mReadCH375Data( DATA_BUFFER );  /* 读取配置描述符数据 */
/* 分析设备描述符和配置描述符，识别 USB 设备，并加载相应的配置 */
```

4、关于 U 盘文件接口

由于以文件格式读写 U 盘的程序量远大于将 U 盘当作存储器的应用，所以，我们目前还提供有关 U 盘文件操作的几种简便方式。

4.1. U 盘文件读写模块（特点：只需要极少的硬件资源，省事，快速应用）

模块中包含 CH375 和一个单片机以及必要的 RAM 缓冲区，系统的主单片机只要将文件名提供给模块（数据流模式可以不需要文件名），就可以读写文件。模块支持 FAT12、FAT16 和 FAT32 文件系统的 U 盘（含闪存盘和外置硬盘，下同），支持搜索文件、读写文件、查询文件、删除文件、新建文件、新建子目录等操作以及与计算机通讯。

如果以扇区为单位读写 U 盘中的文件，那么外部单片机系统需要具备以下硬件资源：不少于 300 字节的程序空间，不少于 520 字节的随机存储器 RAM，部分 U 盘需要不少于 2.2K 字节的 RAM。

如果以字节为单位读写 U 盘中的文件，那么外部单片机系统只需要具备以下硬件资源：不少于 200 字节的程序空间，不少于 16 字节的随机存储器 RAM，当然 RAM 多些更佳。

模块对外提供 8 位被动并口或者异步串口或者其它定制接口，可以通过计算机在线配置或者跳线选择并口或者串口连接，其中的串口连接又分为 3 线制、4 线制、4 线制+中断等方式，这些都可以适用于各种常用的单片机或者 DSP 系统，以较少的工作量让单片机系统支持 U 盘移动存储。

如果单片机系统的 RAM 少于 600 字节，那么只能使用 U 盘文件读写模块。

零售的 U 盘文件读写模块内置了 USB 升级固件，可以由用户自己在计算机上通过普通的 USB 连线升级模块中的程序，也可以设定模块的对外接口方式、设定串口的通讯波特率等。

最新的详细说明请参考 U 盘模块的详细使用说明 CH375HM.PDF。

4.2. U 盘文件级子程序库（特点：批量时硬件成本较低，应用灵活）

由于 CH375 已经内置了 USB 海量存储设备的部分协议，所以外部程序比较精简，硬件上只需要在原单片机系统中增加一个 CH375 芯片，综合成本较低。使用 U 盘文件级子程序库，单片机系统需要具备以下硬件资源：不少于 4KB 到 8KB 的程序空间，不少于 600 字节的随机存储器 RAM（部分 U 盘需要不少于 2200 字节的 RAM），不同的子程序库对内部 RAM 的占用稍有不同。

最新的详细说明请参考 U 盘文件级子程序库说明 CH375HF.PDF。

4.3. U 盘文件级子程序的 C 源程序（特点：可以结合实际应用做进一步优化）

源程序可以适用于常用的 8 位、16 位、32 位等多种单片机和 DSP，特别针对硬件资源有限的中低端单片机进行优化。目前，源程序只针对批量用户提供免费使用授权，本公司保留相关一切著作权。目前可以提供的普及版源程序，支持 FAT16 文件系统，支持文件读写、删除、新建等，经过简单优化。

另外，还可以提供基于子程序库的 U 盘文件读写模块的 C 语言源程序，不含库的源程序。

4.4. 免费试用版：U 盘文件操作的 C 源程序

该源程序从 PC 机的其它操作系统移植过来，由于占用 ROM 程序空间和 RAM 数据空间较多，只能适用于硬件资源较多并且带硬件乘法器的 16 位或者 32 位高速单片机，如果用于 8 位的 MCS51 单片机，那么 U 盘文件读写速度可能比前面的正式版慢一倍。

该源程序只支持 FAT16 文件系统，适用于 CH375 芯片，源程序中注释较少，因为该程序结构与公司的正式版有较大区别，所以公司只能提供非常有限的技术支持。

该源程序可以在购买 CH375 芯片或者 CH375 评估板时附送。

5、其它说明

5.1. 硬件方面

⑴ 如果用频率计测量 CH375 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，普通晶体可以满足晶体 X1 的 0.4‰精度要求。如果电源电压为 3.3V，建议将 XI 引脚的电容 C1 容量选用小些（例如 20pF 甚至 15pF），或者用有源晶振或者外部振荡电路为 CH375 的 XI 引脚提供时钟。

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 以及电容 C1、C2 应该尽量靠近 CH375，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。USB 数据线 D+和 D-应该平行布线，长度保持差不多，两侧可以环绕接地线或者敷铜，更详细的说明可以参考 USB 规范。

⑶ 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，为 CH375 的 USB 引脚 D+和 D-提供进一步的保护。详细说明可以参考 CH375 电路设计注意事项 README.PDF 文档。

⑷ 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 CH375 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 CH375 的影响。

⑸ 如果需要减小电流消耗，可以在空闲时使 CH375 进入低功耗睡眠挂起状态，当有 USB 设备插拔时 CH375 会自动唤醒。在 CH375 睡眠期间，应该使 CH375 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。部分型号的 CH375 芯片不支持睡眠功能。

5.2. 单片机程序方面

⑴ 在单片机程序设计中，向 CH375 发出命令后应该延时 1.5 微秒（CH375A/V/S 芯片需 2 微秒）再读写数据，连续读写数据之间要有 0.6 微秒（CH375A/V/S 芯片需 1 微秒）以上的间隔时间。有些命令需要几微秒的执行时间，尤其在采用较高时钟速度的单片机时特别要注意，具体参数可以参考 CH375 芯片手册。

⑵ 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。

⑶ 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，以“单 DPTR 和 P2+R0 复制”方式传输文件数据（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 100KB 到 200KB，文件越零碎，传输速度越慢。

5.3. 单片机读写 U 盘

⑴ 应用建议：用量不多的用户可以使用 U 盘读写模块，综合成本较低；一般用户可以使用 U 盘文件级子程序库，硬件成本远低于 U 盘读写模块，只需要在原有系统中增加一只 CH375 芯片。

⑵ 以扇区为单位的文件读写子程序，速度较快，操作效率高，但是如果文件长度不是扇区的整数倍，那么就需要自行考虑文件长度的问题。

⑶ 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，但是速度比以扇区为单位的文件读写慢，并且频繁地向 U 盘中的文件写入零碎的数据，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）。另外，每次可以读写的最大字节长度还与 mCmdParam 结构的总长度有关。

⑷ 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

⑸ 关于如何节约 CH375 子程序库所占用的 RAM

CH375 子程序库总共占用约 600 个字节的 RAM（适用于大多数 U 盘，如需支持所有 U 盘则可能需要 2150 字节甚至 4200 字节），其中对于 MCS51 单片机还分为：

外部 RAM 需 512 字节（部分 U 盘需 2K 甚至 4K 字节），用于磁盘数据缓冲 pDISK_BASE_BUF，
外部 RAM 需 30 字节（概数），用于各种全局变量和局部变量，
内部 RAM 需 52 字节（概数），用于各种全局变量和局部变量，同时提供完全不用内部 RAM 的库。
在调用子程序时需要使用 CmdParam 结构，最少为 16 字节内部 RAM，默认为 30 字节，
合计内部 RAM 需要最少约 68 字节，默认情况下约为 82 字节。

CH375 的子程序库经过多次优化，基本上已经很难再节约 RAM 了，对于 MCS51 单片机的子程序库，已经部分采用汇编语言优化，既优化运行速度，又优化 RAM 占用数

关于如何节约 CmdParam 结构所占用的 RAM，对于 MCS51 是内部 idata_RAM

当没有调用 CH375 子程序库时或调用返回后，CmdParam 结构可以用于其它任何用途，从而可以与其它应用程序共用 CmdParam 结构所占用的几十个字节的 RAM。这个方法完全没有副作用，实际上 CmdParam 所占用的 RAM 长度还可以自行定义。

关于如何节约 pDISK_BASE_BUF 缓冲区所占用的 RAM，对于 MCS51 是外部 xdata_RAM

由于部分 U 盘的扇区较大，所以为了支持所有 U 盘，pDISK_BASE_BUF 必须指向一个 2K 甚至 4K 字节的缓冲区，如果仅需支持大部分 U 盘，那么 pDISK_BASE_BUF 可以仅指向一个 512 字节的缓冲区。CH375 子程序库所用的 pDISK_BASE_BUF 缓冲区可以用于其它任何用途，但是在用完后必须调用 CH375DirtyBuffer 通知子程序库“缓冲区被用过”，以免子程序库仍然以为其中数据有效，通过这种简单的方法，就可以让 512 字节甚至 4K 字节的 RAM 与其它应用程序共用。副作用是降低效率，当 CH375 子程序库在需要磁盘数据时必须重新从 U 盘中读取，而无法直接使用缓冲区中的数据，因为缓冲区中的数据已经无效。如果单片机的其它程序模块已经分配好一个缓冲区并且不是一直在使用，那么无需单独为 CH375 子程序库分配 DISK_BASE_BUF 缓冲区，而是将缓冲区指针 pDISK_BASE_BUF 直接指向其它程序模块的缓冲区，以使 CH375 子程序库与其它程序模块实现分时合用。

关于如何节约 CH375 子程序库占用的所有 RAM，对于 MCS51 则包含内部 RAM 和外部 RAM

在没有调用 CH375 子程序库时或调用返回后，其它应用程序可以使用子程序库占用的任何 RAM，但是在用完后，必须将全局变量 CH375DiskStatus 置为 0(也就是 DISK_UNKNOWN)，以通知 CH375 子程序库完全重来，这种方法可以节约子程序库占用的所有 RAM，但是效率较低，将 CH375DiskStatus 清零会使 CH375 子程序库完全重新分析 U 盘的文件系统，可能需要多花几十毫秒甚至数百毫秒时间。另外，也可以调用 CH375SaveVariable 备份所有变量。

⑹ 下表是单片机通过 U 盘文件级子程序库读写 U 盘文件时的速度，是在以 30K 大数据块读取 5MB 大文件时测得的，在小数据块读取时的速度要下降为 70%到 95%。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%，在小数据块写入时的速度通常再下降一半。

单片机硬件类型	文件读写的数据复制方式	读速度/字节每秒
工作于 24MHz 系统时钟的标准 12 时钟 MCS51 单片机（非 12 时钟需参考计算）	常规方法：单 DPTR 复制	108K
	依赖新硬件：双 DPTR 复制	160K
	优化的单 DPTR 和 P2+R0 复制	203K
工作于 40MHz 系统时钟的标准 12 时钟 MCS51 单片机（非 12 时钟需参考计算）	常规方法：单 DPTR 复制	160K
	依赖新硬件：双 DPTR 复制	225K
	优化的单 DPTR 和 P2+R0 复制	280K
大多数 8 位单片机	手工汇编优化可以提高速度	50K 到 500K
大多数 16 位或 32 位单片机	通常不需要手工汇编优化	200K 到 600K