Release Notes for Microchip Memory Disk Drive File System

Version 1.1.2
June 25, 2008

1. Description

This library is intended to provide an interface to file systems compatible with ISO/IEC specification 9293 (commonly referred to as FAT12 and FAT16). This library includes four different physical interface files: one for SecureDigital card interface using the SPI module, one for CompactFlash card interface using manual bit toggling, one for CompactFlash card interface using the Parallel Master Port module included on several PIC24/PIC32 microcontrollers, and one template interface file that can be modified by the user to create a custom interface layer to an unsupported device..

2. Changes In This Release

From version 1.1.1

a. Fixed a bug that prevented the allocation of new clusters to the root directory in FAT32 implementations.
b. Fixed a bug that prevented writing more than one cluster's worth of file entries to the root directory in FAT16/FAT12 implementations.
c. Fixed a bug that returned an incorrect date for directory entries located in the first directory entry after a cluster boundary of a FAT32 root directory.
d. Fixed a bug with FSrename that would cause the function to improperly fail if the directory entries in the current working directory (or previous directory, when renaming the CWD) completely filled a cluster (and no data clusters were allocated to the directory after that).

From version 1.1.0

a. Fixed a bug with the PIC24 clock divider that was causing the interface to run more slowly than intended.
b. Added support for PIC32 microcontrollers.

From version 1.01

a. Added support for FAT32. To enable this functionality, make sure the SUPPORT_FAT32 macro is uncommented in FSconfig.h.
b. Added functions to provide support for the USB Mass Storage Host code.
c. Moved pin and hardware definitions from physical interface files to HardwareProfiles.h.
d. Created function pointers for functions that vary between interface files. These are located in FSconfig.h.

e. Moved macros to select the correct physical layer to HardwareProfiles.h.
f. Modified the SD-SPI physical layer to ensure that communication speed during startup falls between 100 kHz and 400 kHz
g. Created a new example project: MDD File System-PIC24-SD Data Logger. This project contains code for a shell-style program based on the USB Thumb-drive shell demonstrated in Application Note 1145.
h. Decreased the delay in the SD-SPI media initialization from 100 ms to 1 ms.
i. Added the ability to change directories when writes are disabled.

From version 1.0

a. FindFirst and FindNext will now return the create time/data in the timestamp field of a SearchRec object when they return values for a directory.
b. Corrects a bug in the FindEmptyCluster function when searching for files beyond the end of a storage device.
c. Automatically aligns buffers for 16-bit architectures.
d. For the SPI interface, prescaler divides will now be determined dynamically based on the system clock speed defined in FSconfig.h.
e. The DiskMount, LoadMBR, LoadBootSector, and FSFormat functions, as well as the gDiskData, gFATBuffer, and gDataBuffer structures are now located in FSIO.c instead of in the interface files.
f. The SectorRead function will now do a dummy read of the sector and discard the data if it is called with NULL as the data pointer.
g. Replaced the device initialization code in the FSFormat function with calls to InitIO and MediaInitialize.
h. The MediaDetect function is not de-bounced. In order to determine that a device is available, you must call MediaDetect, wait for an appropriate amount of time, and then call it again.
i. The sample linker script in the MDD File System-PIC18-CF-DynMem-UserDefClock project has been modified. Previously, several databanks were merged together; this caused an issue accessing variables that spanned multiple data banks. C18 only allows users to access variables like these using pointers.
j. Added a new user function. The FSrename function will allow the user to rename files and directories. A version that accepts a ROM filename is available for PIC18 (FSrenamepgm). The API is:

```
Function:         int FSrename (const char *fileName, FSFILE * fo)
PreCondition:     None
Input:            fileName    - The new name of the file
                  fo          - The file to rename
Output:           int         - Returns 0 if success, -1 otherwise
Side Effects:     None
Overview:         Change the name of a file or directory
Note:             This function will change the name of the current
                  working directory if 'fo' equals NULL.
```

3. Known Issues

a. This implementation does not support long file names. When using the FSremove or FSremovepgm functions on a file with long file names, the file's FAT entries and short

name directory entry will be deleted successfully, but any long file name entries will not be removed.

4. Compiler Version Used

This library was compiled using MPLAB C18 v.3.20, MPLAB C30 v.3.10, and MPLAB C32 v1.0 C compliers.

5. Memory Size

Unoptimized memory usage for the file interface library using the SD-SPI physical layer is given in Table 1. 512 bytes of data memory are used for the data buffer, and an additional 512 are used for the file allocation table buffer. Additional data memory will be needed based on the number of files opened by the user at once. The default data memory values provided include space for three files opened in static allocation mode. The C18 data memory value includes a 512 byte stack. The first row of the table indicates the smallest amount of memory that the library will use (for read-only mode), and each subsequent row indicates the increase in memory caused by enabling other functionality. Optimized and unoptimized totals for program and data memory with all functions enabled are listed after the table. This data was compiled while allowing two file objects to be opened simultaneously.

Table 1: Memory Usage (Unoptimized)

Functions Included	Program Memory (C18)	Data Memory (C18)	Program Memory (C30)	Data Memory (C30)	Program Memory (C32)	Data Memory (C32)
All extra functions disabled (read only mode)	24701 bytes	1872 bytes	12714 bytes	1324 bytes	22708 bytes	3056 bytes
Read only mode with directory support	+7390 bytes	+75 bytes	+3651 bytes	+78 bytes	+4748 bytes	+88 bytes
File Search enabled	+3728 bytes	+0 bytes	+1638 bytes	+0 bytes	+2288 bytes	+0 bytes
Write enabled	+16086 bytes	+0 bytes	+8307 bytes	+0 bytes	+4844 bytes	+0 bytes
Format enabled (Write must be enabled)	+4778 bytes	+0 bytes	+2571 bytes	+0 bytes	+3624 bytes	+0 bytes
Directories enabled (With writes enabled)	+17001 bytes	+90 bytes	+8694 bytes	+78 bytes	+11560 bytes	+88 bytes
FSfprintf enabled	+17153 bytes	+58 bytes	+4731 bytes	+0 bytes	+8412 bytes	+0 bytes

Functions Included	Program Memory (C18)	Data Memory (C18)	Program Memory (C30)	Data Memory (C30)	Program Memory (C32)	Data Memory (C32)
File Search and Directories enabled	+236 bytes	+0 bytes	+51 bytes	+0 bytes	+7100 bytes	+0 bytes
Pgm functions enabled	+2394 bytes	+0 bytes	N/A	N/A	N/A	N/A

Total memory usage*

C18:

Unoptimized Program memory- 68924 bytes
Unoptimized Data memory- 1962 bytes
Optimized Program memory- 36660 bytes
Optimized Data Memory- 1962 bytes

C30:

Unoptimized Program memory- 38703 bytes
Unoptimized Data memory- 1402 bytes
Optimized Program memory- 23409 bytes
Optimized Data memory- 1402 bytes

C32:

Unoptimized Program memory- 59552 bytes
Unoptimized Data memory- 3146 bytes
Optimized Program memory- 36604 bytes
Optimized Data memory- 3146 bytes

*Note: C18 total memory usage does not include FSfprintf functionality. Since FSfprintf requires integer promotion to be enabled, using it greatly increases the code size of all functions.

6. More Information

More detailed information about the operation of this library is available in Application Note 1045, available from www.microchip.com.