

CSM FAT File System



- ❑ **Support of the typical MS-DOS features:**
 - 12- and 16-Bit FATs
 - Subdirectories
 - File names in 8.3-format
 - File attributes, date and time
- ❑ **Portable by source code according to ANSI C**
- ❑ **For 8-, 16- and 32-bit processors and microcontrollers**
- ❑ **Binary representation in INTEL- or MOTOROLA-format**
- ❑ **Efficient usage of memory**
- ❑ **Perfect for ATA PC Cards, CompactFlash Cards, MultiMediaCards, SD Memory Cards and SRAM Cards**

Introduction

The file system introduced by **MS-DOS** is very common on PCs and available under Windows and various other operating systems, too. Therefore it is often a suitable storage format for custom applications and for the data exchange between these systems and the rest of the world.

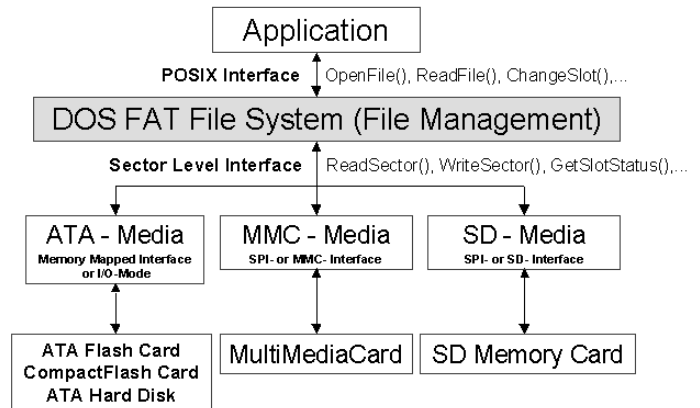
Suitable storage media are **ATA Cards, CompactFlash Cards, MultiMediaCards, SD Memory Cards** or **SRAM Cards**.

Besides its main application the target system must contain additional software layers to maintain the storage format (file system) and to access the storage media (low level functions).

With the file system layer, the developer can fall back on the **CSM FAT File System**. Besides other things this development kit contains a portable and well documented **ANSI-C** source code implementing a MS-DOS V6.x compatible file system.

With **CSM FAT File System** the developer can integrate a well-trying and reliable module into his own application.

Software Architecture



Development Kit

The **CSM FAT File System – Development Kit** includes a complete test system for the following platform:

- **Standard PC with Centronics Interface**
- **Operating system MS-DOS V 6.x, Win 95/98/Me or WIN NT/2000**
- **Memory Card Drive CSM OmniDrive Professional**
- **Development system Microsoft Visual C++ 6.0 (WIN32) or Borland C V3.1(16-Bit)**

For this test system the low level functions to access the Memory Card in the OmniDrive are included as a module.

Some **sample applications** including source code and configuration files for compilation are provided in the package.

The **CSM FAT File System** is integrated into the sample applications as a source code module.

For **CSM FAT File System** a project specific license will be granted. The particular application can be shipped in any number without additional license costs. CSM expects that the customer signs a **license agreement** before delivery.



Features

Comfortable, file oriented **interface to the application** (following POSIX).

- Create, open and close of files:
CreateFile(), OpenFile(), CloseFile()
- Read or write from/to files:
ReadFile(), WriteFile()
- Positioning in a file:
SeekFile(), FlushFile()
- Renaming and deleting files:
RenameFile(), DeleteFile()
- Working with file attributes:
GetFileMode(), SetFileMode(), GetFileDateAndTime()
- Directory maintenance: MakeDir(), ChangeDir(), RemoveDir()
- Directory read: ReadDir()
- Select slot, format the PC Card:
ChangeSlot(), FormatVolume()

Highly **portable** source code (suitable for use in microcontroller application):

- **Byte order** (Big-Endian or Little-Endian) selectable by conditional compilation

Usage

CSM FAT File System is basically suitable for a wide spectrum of systems:

- 8-bit, 16-bit and 32-bit **processors and microcontrollers**
 - ✓ 8051 and derivatives
 - ✓ Intel x86 (8088, 8086, 286, 386, 486, Pentium) and compatible
 - ✓ Siemens C16x-family
 - ✓ Motorola 680x0, 683xx and other
 - ✓ ...
- **Economical usage of memory**
 - ✓ approx. 4.5 KB RAM
 - ✓ approx. 5 KB to 10 KB code
- diverse PC Card **hardware interfaces**
 - ✓ PC Card controller
 - ✓ Mapped into memory space
 - ✓ Mapped into I/O space
 - ✓ PC Card drive
 - ✓ ...
- Various **PC-Card types**
 - ✓ ATA interface (ATA Flash Card, CompactFlash Card, Hard Disk)
 - ✓ SPI or MMC interface (MultiMediaCard)
 - ✓ SPI or SD interface (SD Memory Card)
 - ✓ Linear addressing (SRAM Card)

		CSM FAT File System Development Kit		User System		
Platform:		Standard PC with Centronics Interface, MS-DOS V6.x, WIN 95/98/Me/ NT/2000, Microsoft Visual C++ 6.0 (WIN32) or Borland C V3.1(16-Bit)		project specific Hardware		
		Description	Source	EXE	Description	Action
Application:		sample-application (_COPY, _FORMAT)	✓	✓	custom specific application software of the system	in responsibility of the developer
Interface:		file-oriented ("POSIX")		file-oriented ("POSIX")		
CSM FAT File System:		source code as delivered	✓	✓	adapted source code	adaptation of the source code to the hardware platform
Interface:		sector-oriented		sector-oriented		
Low-Level-Functions:		module for CSM OmniDrive Professional		✓	low level functions access routines	not included, to do by the developer
Hardware:		CSM OmniDrive Professional		project specific		

Structure

CSM FAT File System introduces a triple layer software model:

- Application (upper layer)
- **CSM FAT File System** (middle layer)
- Low level functions (lower layer)

The **CSM FAT File System – Development Kit** includes a reference realization for the standard platform mentioned above:

- Sample application in C-source code and executable as EXE: `_COPY`, `_FORMAT`
- **CSM FAT File System** as ANSI-C source code, for the standard platform can be used without any change
- Low level functions as module for CSM *OmniDrive* Professional (source code not necessary)

For applications with **CSM FAT File System** on other hardware platforms (e.g. microcontroller systems, process control, industrial computers etc.) the developer can integrate the source code of **CSM FAT File System** in his software. This may have the following structure:

- Custom specific application software of the system
- **CSM FAT File System** adapted to the specific platform by the developer
- Low level functions with sector oriented interface and access routines to the Memory Card, has to be implemented according to the requirements of the target system.

CSM FAT File System needs few, precisely specified low level functions:

- Read/write of 512 byte sectors: `ReadSector()`, `WriteSector()`
- PC Card inserted? Card changed?, Card write protected?: `GetSlotStatus()`
- Memory capacity of the card: `GetSizeOfCard()`
- Date and time (for file date and time): `DOSDate()`, `DOSTime()`

Adaptation to the Target System

During software implementation for a specific target system (e.g. embedded system with μ -Controller) the following tasks have to be done:

➤ **Providing access routines to the Memory Card.**

These routines depend on the hardware design. It must be decided which types of Memory Cards (ATA Cards, CompactFlash Cards, MultiMediaCards, SD Memory Cards or SRAM Cards) are to be used.

The **CSM FAT File System** has no restrictions to these routines but it is recommended to pay attention to the PC Card Standard.

➤ **Realization of the low level functions according to the interface definition of the CSM FAT File System.**

These functions are using the access routines (see above). With a sector oriented interface the low level functions separate all hardware dependent and Memory Card specific details from the **CSM FAT File System**.

➤ **Adapting the CSM FAT File System.**

The source code is optimized for high portability. But due to the many imaginable hardware platforms and system environments, the need for specific adaptations cannot be ruled out. The documentation provides an optimal support for that task.

The following points deserve special attention:

➤ **Alignment of data:**

The layout of the FAT control structures is fixed by the definitions of MS-DOS. This has to be considered at some data structures of **CSM FAT File System**.

➤ **Access to data structures:**

Some systems limit data access to specific address boundaries (word, dword, ..).

➤ **Multitasking environment:**

It has to be checked individually if the target system requirements allow the usage of **CSM FAT File System**.

The implementation of locking mechanisms to protect against concurrent access may be required.

➤ **Real time environment:**

It has to be checked individually if the target system requirements allow the usage of the **CSM FAT File System**.

➤ **Integration with application:**

Using of the file oriented interface provided by the target application.

The **integration** of **CSM FAT File System** into custom specific software for a specific hardware platform requires a certain amount of **engineering know-how** (like any other responsible development job).

The **CSM FAT File System - Development Kit** offers a valuable knowledge about the file system by providing documentation and source code.

Application Training

The purchase of **CSM FAT File System** includes a half-day training for one employee, where important information is provided for both usage of **CSM FAT File System** and interface of used storage medium. The training is individual for the engineer and can be adapted to the intended application (e.g. type of microcontroller).

The training takes place at CSM Filderstadt.

Shipping Contents CSM FAT File System:

- **CSM FAT File System** as ANSI-C source code (consisting of files `File.c`, `File.h` and `My-Defs.h`).
The source code includes detailed and useful comments.
- **Documentation** (reference manual): overview, description of the included files, reference of the provided functions, variables and data structures, hints for adaptation to various configurations and processors respectively microcontrollers.
- **Object module** (lower layer) for the PC Card drive CSM OmniDrive for PC based systems.
- **Example applications** in C source code.
- **Project- and configuration files** particular for Microsoft Visual C++ 6.0 (WIN32) or Borland C from V3.1(16-Bit).
- **Half-day application training at CSM**
User specific training, information for usage of DOS FAT File System, hints for design-in of storage medium.

CSM GmbH, Raiffeisenstr. 34, D-70794 Filderstadt

Tel.: +49 711 77964 20 Fax: +49 711 77964 40

mailto: info@csm.de http://www.csm.de

All trademarks mentioned are in property of their respective owners. This document is subject to change without notice.