fat.h

Go to the documentation of this file.

```
/*!     \file include/kernel/fat.h
 *      \brief  FAT file system header.
 *      \author
 *              Matteo Chesi <dalamar@inwind.it>
 *              Rudy Manganelli <feller@libero.it>
 *              Andrea Righi <drizzt@inwind.it>
 *      \note Copyright (&copy;) 2003
 *              Matteo Chesi <dalamar@inwind.it>
 *              Rudy Manganelli <feller@libero.it>
 *              Andrea Righi <drizzt@inwind.it>
 *      \date Last update: 2003-11-09 by Andrea Righi.
 */

#ifndef FAT12_H
#define FAT12_H

/** \ingroup FileSystem
 *  \defgroup FSFAT12 FAT-12
 *  The FAT-12 file system.
 *  @{
 */

#define FAT_PHYS_SIZE           9
#define FAT_BOOT_SECTOR         0
#define FAT_SECTOR_SIZE         512

#define EOF_FAT12       0xFF8
#define EOF_FAT16       0xFFF8
#define EOF_FAT32       0xFFFFFF8

// Stuctures                                                            //
// Boot sector                                                          //
typedef struct bootsect
{
        byte Jump[3];
        unsigned char Name[8];
        word BytesPerSector;
        byte SectorsPerCluster;
        word ReservedSectors;
        byte Fats;
        word RootDirectoryEntries;
        word LogicalSectors;
        byte MediumDescriptorByte;
        word SectorsPerFat;
        word SectorsPerTrack;
        word Heads;
        word HiddenSectors;
        byte code[482];
} __attribute__ ((packed)) bootsect_t;

// Physical FAT structure (into the disk)                               //
typedef struct FAT12
{
        byte data[FAT_SECTOR_SIZE*FAT_PHYS_SIZE];

}  __attribute__ ((packed)) FAT12_t;

// Logical transposition of the FAT structure                           //
typedef struct logical_FAT12
```

```
00060 {
00061         word data[3072];
00062 } __attribute__ ((packed)) logical_FAT12_t;
00063
00064 // Sector buffer                                                    //
00065 typedef struct sector
00066 {
00067         byte data[FAT_SECTOR_SIZE];
00068
00069 } __attribute__ ((packed)) sector_t;
00070
00071 // File entry type                                                  //
00072 typedef struct FileEntry
00073 {
00074         unsigned char Name[8];
00075         unsigned char Extension[3];
00076         byte Attribute;
00077         byte Reserved[10];
00078         word Time;
00079         word Date;
00080         word StartCluster;
00081         dword FileLength;
00082
00083 } __attribute__ ((packed)) FileEntry_t;
00084
00085 // Date type                                                        //
00086 typedef struct date
00087 {
00088         int year;
00089         int month;
00090         int day;
00091
00092 } __attribute__ ((packed)) date_t;
00093
00094 // Time type                                                        //
00095 typedef struct fat_time
00096 {
00097         int hour;
00098         int minute;
00099         int second;
00100
00101 } __attribute__ ((packed)) fat_time_t;
00102
00103 // Attribute type //
00104 typedef struct attrib
00105 {
00106         bool RW;
00107         bool Hidden;
00108         bool System;
00109         bool Label;
00110         bool Directory;
00111         bool Archived;
00112         byte Reserved;
00113
00114 } __attribute__ ((packed)) attrib_t;
00115
00116 // Sector Directory type //
00117 typedef struct SectorDir
00118 {
00119         FileEntry_t Entry[FAT_SECTOR_SIZE/sizeof(FileEntry_t)];
00120
00121 } __attribute__ ((packed)) SectorDir_t;
00122
00123 // --- Prototypes ------------------------------------------------- //
00124
00125 bool Read_FAT();
00126 bool load_file(char *stringa, byte *buffer);
00127 int get_file_size(char *file_name);
00128 char *pwd();
```

```
00129 void ls();
00130 bool cd(char *new_path);
00131 bool cat(char *stringa);
00132 bool rm(char *filename);
00133
00134 /** @} */ // end of FSFAT12
00135
00136 #endif
```
