

Pyxis File Allocation Table (PFAT)

Overview

PFAT creates an 8.2 based File System on your EEPROM. Since you may already be using your EEPROM for other saved data PFAT allows you to use an offset when formatting the drive so your existing data can be left alone.

Definitions

PFAT uses several definitions in its header file to determine how to use the space on the EEPROM. Here we will examine each of these definitions.

EEPROM_STRT – This tells us what address on the EEPROM to start at when creating/using the file system.

EEPROM_SIZE – The overall size of the EEPROM chip. **DO NOT** subtract the size of your *EEPROM_STRT* from the size of your EEPROM chip; this is done automatically.

EEPROM_HEAD – Denotes the size of the EEPROM header. For PFAT this is 8 bytes and stores 'PFAT1000'. We need this so we know that the EEPROM has been formatted and which version of PFAT formatted it.

HEADER_SIZE – Tells PFAT how large the overall header is. 8 bytes are used to store the FAT version and another 12 are used for the volume label; hence the default value is 20 (8 + 12).

FENTRY_SIZE – Contains the size of a FAT entry. Each entry into the FAT uses 12 bytes for the name (8.3), 1 byte reserved, and 2 bytes for the address of the first cluster in the file.

SECTOR_SIZE – Defines the size of each sector in a cluster. You are free to adjust this number as you see fit.

CLUSTR_SIZE – Tells PFAT how many sectors to put into each cluster, again you may adjust this number as you like. You'll find that using larger overall numbers decreases the number of clusters per file but increases the amount of wasted space on smaller files, where as smaller numbers create more overhead for larger files.

CLUSTR_HEAD – Each cluster has a header, the size of that header is defined here. By default PFAT uses only one byte which contains the index of the next cluster or 255 if it is the last cluster in the file. Obviously this limits the maximum size of your files. Using the default settings the largest a file could be is 16,320 bytes (16 bytes per sector * 4 sectors per cluster * 255 clusters).

Public Methods

The PFAT library contains many public methods. You will find the use of each of these methods detailed below.

closeFile

You can have up to four files opened at any one time using PFAT. To free the file pointers you must call closeFile with the handle of the file you wish to close

Parameters

Type	Name	Description
int	handle	Handle to the file you wish to close

Returns

Type	Description
bool	True if successful

createFile

Used to create a new file entry; if a file with that name already exists PFAT will return false

Parameters

Type	Name	Description
char[11]	filename	The name you wish to give the new file
int	size	How much space should be allocated

Returns

Type	Description
bool	True if successful, PFAT returns false if the file exists or there is not enough space

deleteFile

When deleting file PFAT simply switches the address of the first cluster to zero (0) in the file's FAT entry. This allows you to undelete files by searching through the clusters

Parameters

Type	Name	Description
int	fileIndex	The FAT entry index of the file you wish to delete

Returns

Type	Description
bool	True if successful

deleteFile

When deleting file PFAT simply switches the address of the first cluster to zero (0) in the file's FAT entry. This allows you to undelete files by searching through the clusters

Parameters

Type	Name	Description
char[11]	filename	The name of the file you wish to delete

Returns

Type	Description
bool	True if successful

formatDrive

Like any other file system PFAT allows you to format the drive. This must be done at least once for PFAT to function properly

Parameters

Type	Name	Description
char[11]	newLabel	This is the name of the EEPROM drive

Returns

None

fileCount

Returns the number of files currently contained on the PFAT EEPROM drive

Parameters

None

Returns

Type	Description
int	Number of files existant

fileIndexByName

Some methods inside PFAT uses the file's FAT entry ID, this method will retrieve the ID for a specified filename

Parameters

Type	Name	Description
char[12]	filename	The name of the file you want to get an index for

Returns

Type	Description
int	The file's FAT entry ID; returns -1 if file does not exist or error occurs

fileNameByIndex

If you have a file's FAT entry ID you can easily retrieve its name with this function

Parameters

Type	Name	Description
char[12]	filename	This character array will be populated with the filename
int	index	FAT entry ID for the file

Returns

None

fileSize

Retrieves the size of the file

Parameters

Type	Name	Description
int	fileIndex	FAT entry ID for the file

Returns

Type	Description
long	Size of the file; -1 if file does not exist or error occurs

fileSize

Retrieves the size of the file

Parameters

Type	Name	Description
char[11]	filename	The name of the file

Returns

Type	Description
long	Size of the file; -1 if file does not exist or error occurs

fileSizeOnDisk

Total amount of space used in the EEPROM to store the file

Parameters

Type	Name	Description
int	fileIndex	FAT entry ID for the file

Returns

Type	Description
long	Size of the file on disk; -1 if file does not exist or error occurs

fileSizeOnDisk

Total amount of space used in the EEPROM to store the file

Parameters

Type	Name	Description
char[11]	filename	The name of the file

Returns

Type	Description
long	Size of the file on disk; -1 if file does not exist or error occurs

firstFreeClusterIndex

Returns the index of the first cluster on the EEPROM not being used by a file

Parameters

None

Returns

Type	Description
int	Address to the first free cluster; -1 if fails

firstFreeClusterIndex

Returns the index of the first cluster on the EEPROM not being used by a file *after* the index supplied; this is used when writing to files

Parameters

Type	Name	Description
int	excludeIndex	ID of the index to exclude

Returns

Type	Description
int	Address to the first free cluster; -1 if fails

firstFreeFATIndex

Returns the index of the first entry in the FAT header not being used by a file

Parameters

None

Returns

Type	Description
int	Address to the first free entry; -1 if fails

freeSpace

Returns the amount of space currently available on the EEPROM for use

Parameters

None

Returns

Type	Description
int	Available space in bytes; -1 if fails

mountDrive

Mounts the EEPROM drive

Parameters

None

Returns

Type	Description
bool	True if successful

openFile

Opens a file on the EEPROM drive

Parameters

Type	Name	Description
int	fileIndex	FAT entry ID for the file

Returns

Type	Description
int	Handle to the file, 0 – 3 are valid; -1 indicates an error

openFile

Opens a file on the EEPROM drive

Parameters

Type	Name	Description
Char[11]	filename	Name of the file

Returns

Type	Description
int	Handle to the file, 0 – 3 are valid; -1 indicates an error

readByte

Reads a single byte from a file

Parameters

Type	Name	Description
int	handle	Handle of the file to read from
int	data	This will be populated with the byte read

Returns

Type	Description
bool	True if successful

readWord

Reads eight bytes from a file

Parameters

Type	Name	Description
int	handle	Handle of the file to read from
Int[7]	Word	This will be populated with the data read

Returns

Type	Description
bool	True if successful

readDoubleWord

Reads 16 bytes from a file

Parameters

Type	Name	Description
int	handle	Handle of the file to read from
Int[15]	Word	This will be populated with the data read

Returns

Type	Description
bool	True if successful

readQuadWord

Reads 32 bytes from a file

Parameters

Type	Name	Description
int	handle	Handle of the file to read from
Int[31]	Word	This will be populated with the data read

Returns

Type	Description
bool	True if successful

renameDrive

Changes the name of the drive without reformatting it

Parameters

Type	Name	Description
char[11]	newLabel	This is the name of the EEPROM drive

Returns

None

writeBlockToFile

Writes an entire cluster to the EEPROM, the size of this block is defined as `SECTOR_SIZE * CLUSTER_SIZE`

Parameters

Type	Name	Description
int	handle	Handle of the file to write to
int*	dataBlock	Pointer to data array

Returns

Type	Description
bool	True if successful

writeByteToFile

Writes a single byte to the file

Parameters

Type	Name	Description
int	handle	Handle of the file to write to
int	data	Data to write

Returns

Type	Description
bool	True if successful

writeCharsToFile

Writes an array of chars to the EEPROM

Parameters

Type	Name	Description
int	handle	Handle of the file to write to
char*	data	Pointer to data array
int	dataSize	Size of the array (or size to write, which may be smaller than array)

Returns

Type	Description
bool	True if successful

writeDataToFile

Writes an array of chars to the EEPROM

Parameters

Type	Name	Description
int	handle	Handle of the file to write to
int*	data	Pointer to data array
int	dataSize	Size of the array (or size to write, which may be smaller than array)

Returns

Type	Description
bool	True if successful

Private Methods

In addition to the publicly exposed methods, PFAT also uses several private methods. These are explained below,

clusterIndexToAddress

Translates a clusters index ID to a physical address on the EEPROM chip

Parameters

Type	Name	Description
int	index	Cluster Index

Returns

Type	Description
int	Translated address

FATIndexToAddress

Translates a FAT entry ID to a physical address on the EEPROM chip

Parameters

Type	Name	Description
int	index	FAT entry ID

Returns

Type	Description
int	Translated address; -1 if fails

FATIndexFromFileIndex

Gets the FAT entry ID for a file from the file's index

Parameters

Type	Name	Description
int	index	File Index

Returns

Type	Description
int	FAT entry ID; -1 if fails

firstClusterInFile

Returns the cluster index for the first cluster in a file

Parameters

Type	Name	Description
int	fileIndex	Index of the File

Returns

Type	Description
int	Cluster Index

getFreeHandle

Returns the first free file handle

Parameters

None

Returns

Type	Description
int	File Handle; -1 if fails

strEqual

Compares two strings

Parameters

Type	Name	Description
char*	str1	First string
char*	str2	Second string
int	length	Length of string

Returns

Type	Description
bool	Returns True if strings match

EEPROM Read

Reads data from the EEPROM chip

Parameters

Type	Name	Description
int	address	Address on EEPROM

Returns

Type	Description
uint8_t	Value stored at that address

EEPROM Write

Writes a byte to the EEPROM chip

Parameters

Type	Name	Description
int	address	Address on EEPROM
uint8_t	value	Value to write to that address

Returns

None