

# **TFS4 CONFIGURING CLUSTER SIZE**

## ***Application Note***

---

November-2006, Version 1.0



# Copyright Notice

---

Copyright © 2006, Flash Software Group, Samsung Electronics Co., Ltd

All rights reserved.

## Trademarks

TFS4 is a trademark of Memory Division, Samsung Electronics Co., Ltd in Korea and other countries.

## Restrictions on Use and Transfer

No part of this guide may be reproduced in any form or by any means (including electronic storage and retrieval or translation into a foreign language) without prior agreement and written consent from Samsung Electronics Co., Ltd. as governed by Republic of Korea and international copyright laws.

Document confidentiality status: Restricted Distribution

Readers of this document are not granted the right to distribute this document to any third party without prior written agreement from Samsung Electronics Co., Ltd.

Samsung Electronics Co., Ltd may make changes to the contents of this document any time without notice.

THIS DOCUMENT IS INTENDED ONLY TO ASSIST THE READER IN THE USE OF THE PRODUCT. SAMSUNG ELECTRONICS Co., Ltd. SHALL NOT BE LIABLE FOR ANY LOSS OR DAMAGE ARISING FROM THE USE OF ANY INFORMATION IN THIS DOCUMENT. SAMSUNG ELECTRONICS Co., Ltd. ASSUMES NO RESPONSIBILITY FOR ANY ERROR OR OMISSION IN INFORMATION CONTAINED IN THIS DOCUMENT, OR ANY INCORRECT USE OF THE PRODUCT.

## Contact Information

Flash Software Group  
Memory Division  
Samsung Electronics Co., Ltd

**Address:** San #16 Banwol-Dong, Taeon-Eup  
Hwasung-City, Gyeonggi-Do, Korea, 445-701

# 1. Introduction

A file system is typically used by an operating system or program applications to locate, name, organize, and store files.” Files “ are named collection of information in many data formats such as program, data used by a program application, user-centred documents etc. The file system itself consists of files, directories, and information needed to locate and access files. They also include functionality needed to translate requests for files from application programs in to low-level, sector oriented tasks that are understood by a driver and used for communicating with NAND device. Many different types of file system are used in computer systems, each file system providing a different way of organizing and handling data. However, one type of file system, FAT file system is exceptionally common. Because FAT is ubiquitous, most non-Microsoft OS (e.g. Linux, and Apple’s MAC-OS) that have their own OS also support FAT. Most current file systems tend to use rigid architectures that behave same regardless of the underlying storage medium. Accordingly, to facilitate the communication between the FAT File system and the underlying NAND storage medium, Samsung has come out with its Transaction File System (TFS4) developed to function with flash memory.

# 2. Clusters

A FAT storage medium is physically divided into sectors (traditionally 512 bytes each). From the file system’s point of view, the storage medium is linear array of sectors, starting from the first sector, sector 0. The lowermost sectors of the storage medium contain the basic FAT structures, including the FAT table. These are followed by the rest of the storage medium, which contains all the file and directory data and available free space. This part is divided in to allocation units, also called clusters. An allocation unit is the minimum space that can be allocated to a file or a directory and its size is fixed throughout the storage medium. The size of the allocation unit is the multiple of the sector size e.g. 4Kbytes(=8 sectors). The file system does not use an allocation table to store file system’s meta information. Physical sectors are logically organized into two regions: a data region for storing data such as files, directories, attributes, etc; and a meta data area of the flash medium to store meta information.

# 3. FAT

The FAT is a table that indicates the status of each allocation unit. A FAT entry may show that an allocation unit is free space, or it may show that it’s allocated to a file (though it will not show to which file). In the later case the FAT entry also indicates what the next allocation unit for the file is, or indicates that this allocation unit is the last allocation unit of the file. This organization leads to a file having a FAT chain: a list of chained entries in the file allocation table showing which allocation units belong to the file and in which order. The number of bits in each entry is called the FAT size and is one of three values, 12, 16 or 32. The FAT type is determined by the operating system based upon cluster count it takes to represent the volume. If the volume requires more than 4096 but less than 65536, then a 16-bit FAT is used. Otherwise, a 32-bit FAT is used for the device.

**Table 1. Clusters per FAT**

| FAT Size in bits | Number of Clusters |
|------------------|--------------------|
| 12               | 4096               |
| 16               | 65536              |
| 32               | 4,294,967,296      |

**Table 2. Volume Size**

| Cluster Size (Bytes) | FAT 12  | FAT 16 | FAT 32  |
|----------------------|---------|--------|---------|
| 1024 (1K)            | 4.1MB   | 67MB   | 4.39TB  |
| 2048 (2K)            | 8.2MB   | 134MB  | 8.78TB  |
| 4096 (4K)            | 16.7MB  | 268MB  | 17.59TB |
| 8192(8K)             | 33.4MB  | 536MB  | 35.18TB |
| 16384(16K)           | 66.8MB  | 1.07GB | 70.36TB |
| 32768(32K)           | 134.2MB | 2.14GB | 140.7TB |

Table 2 illustrates the maximum size of volume give the cluster size and FAT type

The FATxx volume is divided into four areas:

- The boot record
- The File Allocation Tables
- The root directory
- The data area

The boot record is the first sector of a FAT16 volume, and the first 3 sectors of a FAT32 volume. Two important things that are stored in the boot record is the size of each sector usually 512 bytes and the total number of sectors. With this information we can tell how much data the volume will hold. If the volume is bootable, then the first sector of the boot record also contains the code required to enter the file system and boot the OS.

Other information that the boot record holds include:

- The name of the operating system that formatted it.
- Sectors per cluster
- The maximum number of root directory entries you can have for the volume
- The volume name
- The serial number.

The boot sector is vital to be able to read the volume. If this goes the operating system will not know what type/size volume it is dealing with, because of this fact the FAT32 file system stores a copy of the boot record.

The File Allocation Table is a series of addresses that is accessed as a lookup table to see which cluster comes next, when loading a file or traversing a directory. When it's stored on volume, a file is broken up

into cluster size pieces and then written to the data area. Provided you can track these fragments of the file then the file does not have to be written in to consecutive clusters, for instance a file that splits into 3 fragments need not be stored in clusters 5, 6 and 7, it could go into 5, 9 and 36 instead. The file allocation table facilitates this action, and more besides.

Basically, a file allocation table is just a load of numbers, no filenames, no attributes all those things are stored elsewhere. For every cluster on the disk there is an entry in the file allocation table which occupies the number of bits that we are using (12, 16 or 32).

These numbers hold the status of each and every cluster, for instance if the cluster is free for use then the FAT entry for it will hold 0, if a cluster is bad (cannot be used) then this too will be indicated.

The file allocation table contains the following types of information about each cluster on the volume:

- Unused (0x0000)
- Cluster in use by a file
- Bad cluster (0xFFFF7)
- Last cluster in a file (0xFFFF8-0xFFFF)

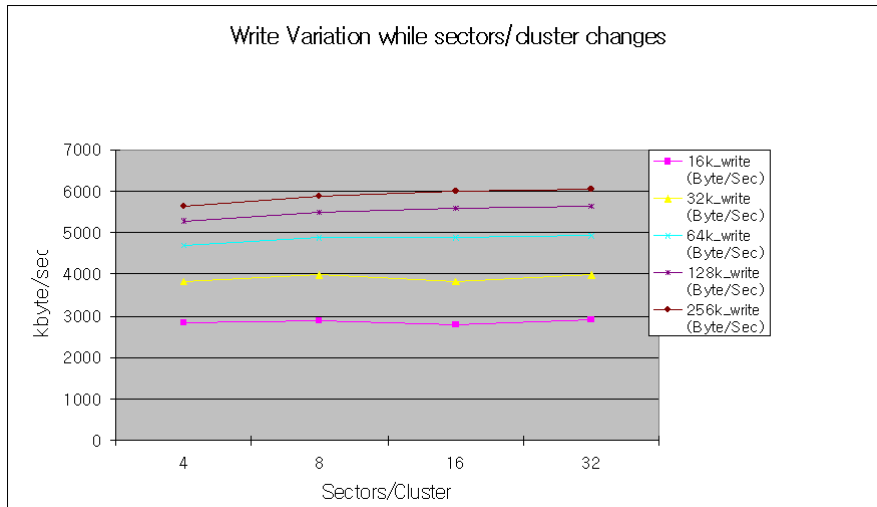
There is no organization to the FAT folder structure, and files are given the first available location on the volume. The starting cluster number is the address of the first cluster used by the file. Each cluster contains a pointer to the next cluster in the file, or an indication (0xFFFF) that this cluster is the end of the file.

The root directory is fixed in length and always located at the start of the volume (after the FAT) in FAT12 and FAT16 volumes, but FAT32 treats the root directory as just another cluster chain in the data area. However, even in FAT32 volumes, the root directory will typically follow immediately after the two FATs.

The data area fills the rest of the volume, and is divided into clusters; it is here that the file data is stored. Subdirectories are files with a particular structure that is understood by the file system, and are marked as being directories rather than files by setting the "directory" attribute bit in the directory entry that points to it.

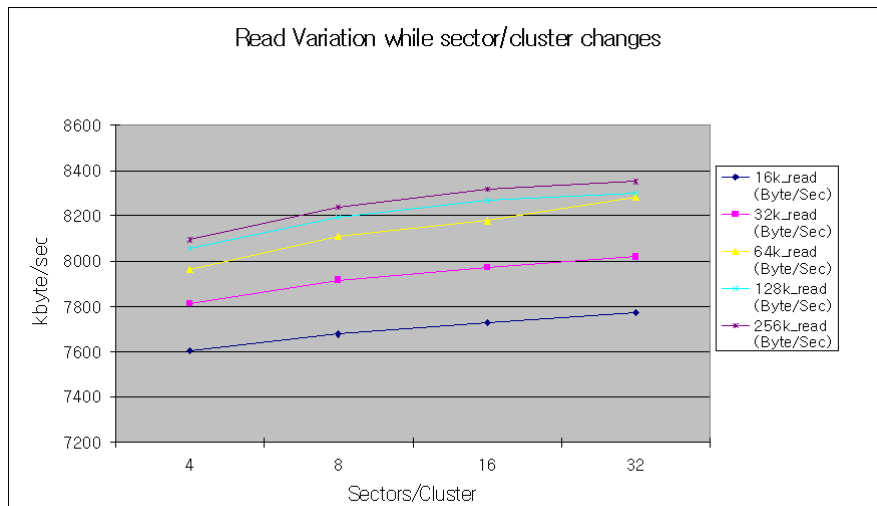
## 4. Cluster Size Effect

The below example illustrates variation of read/write performance with changes in the cluster size for a FAT 16 File System.



**Figure 1. Write Performance**

As shown in Fig 1, the write performance for a 16KB or 32KB data write, shows minimal improvement but write performance for a 54K,128K, 256K data write shows better performance with increase in the cluster size.



**Figure 2. Read Performance**

The read performance improves with increase in cluster size for all 16KB,32KB,54KB,128KB,256KB data read.