

TFS4 FAST SEEK

Technical Paper

November-2007, 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

Introduction

Data is generally stored on a disk in the form of data clusters. A cluster represents several continuous physical sectors of the disk. The data clusters of each file have a logical sequence within the file. Samsung's TFS4 FAT file system uses a linked list structure of file allocation table (FAT) to manage the cluster chain to access files. A cluster chain is a linked list of the data clusters in a particular file. With linked allocation, the clusters may be scattered anywhere on the disk. The beginning of the cluster chain for each file is provided by directory entry of the file. The rest of the cluster chain are obtained from the FAT.

The read and write operations of a file system allows the programs to access the data and store the data to files. These functions require a handle to a file to be opened for reading and writing respectively, the position in the file indicated by the file pointer to begin reading and writing the data, a memory buffer and a length of the data. Using the file position, the corresponding cluster number is computed by scanning the FAT chain from the starting cluster of the file. After the read/write operation the file pointer is incremented by the number of bytes read/written.

The major problem with linked allocations is that in order to locate the target address for read/write operation, the searching occurs sequentially in a forward direction in a FAT table beginning with the starting cluster number. The file pointer is incremented after each operation to reflect the new position. If an address request happens to be from a previous file pointer location, the file system has to start again from the starting address and then position the pointer to set the target location, since the search can happen only sequentially in forward direction in linked list structure of FAT table. If the file is small, the time required to perform this search may be insignificant. However, for files that contain large data, this is time consuming since search is possible only in the forward direction and not reverse.

Fast Seek Access

In order to solve this problem, Samsung introduced a new file access mechanism - Fast Seek method to reduce the search time by creating a fast seek table in addition to the existing FAT. This mapping table is created during the initial file read/write operation and is updated during subsequent file read/write operation. In this method, the file is viewed as a numbered sequence of intervals from 1 to N.

The table has one cluster number for each interval and is indexed by that interval number. The index is determined by dividing the file position by interval size. Once the cluster number is determined using the Fast Seek table, the search is made sequentially in the FAT table starting from this cluster number. This reduces the search time since the search has not be made from the starting cluster address of the FAT table and thereby improves the file system performance. As a result, a file can be accessed faster by a random and sequential scheme, rather than using only sequential access.

Consider a data access initiated by read request to the file system. Given the offset, the index to the fast seek table is computed and based on the index, the cluster number corresponding to the offset is accessed. Once the cluster number is determined, the actual cluster is determined by sequential search in the FAT table. Thus, the file is accessed faster using a combination of random access and sequential access, rather than only sequential access file. For an 1 MB file size and interval of 100 KB, if the target item to be located is at 450 KB, the file pointer is set to cluster corresponding to the 4th interval and the data corresponding to the byte offset 50 (400-450) within the fourth interval would be used.

Performance Improvement

Consider a test environment of 512 MB MMC card having FAT 16 TFS4 file system with cluster size of 4KB (8 sectors), file size of 400 MB and 100 KB interval.

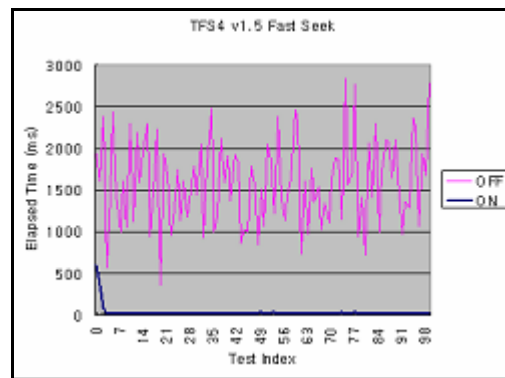


Figure 1. TFS4 Fast Seek Performance

As shown in Fig 1, the X axis indicates the read count and Y axis displays the time for reading. Due to initialization of fast seek table, the estimated time is initially high. Later, the seek time is significantly low in comparison to file system with no fast seek method.