

TFS4 Mount Operation

Technical Paper

November-2007, Version 1.0





Copyright Notice

Copyright © 2007, 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



Abstract

This paper explains the TFS4 mounting function and how the mount operation is executed by the Samsung's TFS4 flash file system. It also explains what happens internally when the mount operation is executed.

Introduction

Mounting is the attaching of a device or partition to the currently accessible file system of a computer. The mount procedure is straightforward. The operating system is given the name of the device, and the location within the file structure at which to attach the file system (or mount point). In TFS4, a partition consisting of physical device name and partition-id is mounted at a mount point. A mount point is a volume on a storage device that is managed by the file system at which the mounted file system will be attached. It becomes the root directory of the added directory tree, and that tree becomes accessible from the volume to which it is mounted (i.e., its mount point).

There are three things that the TFS4 file system needs to know in order to mount a partition or device (like a flash drive):

The device name: such as '/dev/nf0'

The mount point: such as '/a/'

The native file system: such as 'KFATFS' - developed for TFS4

For example, to mount 1st partition of /dev/nf to /a/, type - tfs4_mount("/dev/nf0", "/a/", "KFATFS", 0, NULL);

TFS4 Mounting Process

1. when user calls tfs4_mount function, TFS4 reads the MBR- Master Boot Record is an area on the disk that contains all information about disk partitions. The MBR holds a disk's primary partition table. Each partition table entry is 16 bytes. TFS4 reads the partition table and checks whether the newly mounted partition exists.

2. If such a partition is found, the boot sector of that partition is read into memory to check whether the file system is FAT supported or not. Next, TFS4 determines the FAT entry size - FAT 12 or FAT 16 or FAT 32.

3. After reading from the boot sector, TFS4 initializes the Root Directory and performs secondary functions like log recovery, etc. to use the volume.

Mount Operation

tfs4_mount()

DESCRIPTION

This function mounts a partition onto a file system volume. tfs4_mount() attaches the file system specified by dev to the volume specified by target.

SYNTAX

```
t_int32 tfs4_mount(const t_char* psDev,
const t_char* psTarget, const t_char* psFsType,
t_uint32 dwFlag, const void* pData);
```

PARAMETERS

psDev-The logical device name. Logical device name consists of physical device and partition number. "/dev/nf0" means first partition of physical device registered as "/dev/nf". Partition information can be acquired from tfs4_fdisk_pinfo().

psTarget-Target volume name. You can use volume name from "/a/" to "/z/".

psFsType-This is file system name e.g) "KFATFS"

dwFlagMount flag-

Symbolic constants are provided for dwFlag. Each flag can be calculated and assigned by a logical sum (OR).

TFS4_MOUNT_RDONLY

- It mounts volume as read only.

TFS4_MOUNT_LOG_INIT

- It initializes log files.

TFS4_MOUNT_FAT_MIRROR

- It supports mirroring of a FAT table.

TFS4_MOUNT_NO_LOG

- It mounts a volume without log recovery function.

TFS4 does not create log file and write any log.

Flags for TFS4 hidden protected area

TFS4_MOUNT_HPA: It mounts a volume with hidden area. tfs4_mount() will be failed with this flag when there is no hidden area on the volume.

TFS4_MOUNT_HPA_CREATE: TFS4 creates hidden area when there is no hidden area. It enables hidden area when there is a hidden area without hidden area creation.

data -Current version of TFS4 does not use this parameter. This is for POSIX compatibility.

Just set this parameter as NULL

RETURN VALUE

TFS4_OK-If the function succeeds, the return value is TFS4_OK.

TFS4_EERROR-If the function fails, the return value is TFS4_EERROR.