TECHNICAL
PAPER

Flash File System Selection Guide

KIRK BLUM
MCD SOFTWARE GROUP

August 1996

Order Number: 297665-003

Copies of documents which have an ordering number and are referenced in this document, or other Intel literature, may be obtained from:

Intel Corporation
P.O. Box 7641
Mt. Prospect, IL 60056-7641

or call 1-800-879-4683

CG-041493

CONTENTS

FIGURES

REVISION HISTORY

Number	Description
1.0	Original version.
2.0	Content change to reflect current software Plan of Record.
2.1	Internal release.
2.2	Minor cosmetic changes for inclusion in databook.
2.3	Content changes to match databook version and cosmetic changes for better FaxBack* transmission.
2.4	Minor text revisions and figure additions.
2.5	Content changes and FTL content improvement.
3.0	Structural layout changes only.

1.0 INTRODUCTION

This document provides guidelines for determining the “best fit” file system for use with flash in a particular implementation. Now that flash technology has matured, is widely available, and has become an industry standard, more attention is being given to improving the necessary enabling software. The Flash Translation Layer (FTL) flash driver has emerged as the best flash software solution for most applications, and is currently being supported by several companies including M-Systems, SCM, Datalight and SystemSoft. Intel strongly advocates the use of FTL with flash for most PC and embedded applications, especially those that need to transfer data back to the PC.

However, after discussions with many embedded system developers, Intel realizes that sometimes FTL solutions may not meet all customers’ needs. This document presents FTL, FFS2, and other software solutions, and categorizes them for easy understanding. Additionally, a straightforward file system selection tree is presented as an aid in determining the best implementation for your system.

It is important to understand that each file system has its own strengths and weaknesses, and that there is no solution that meets every design’s needs. Helping you determine which file system best meets your need is the goal of this document.

Intel is committed to supporting our customers. If you are interested in flash, want to know about flash, or want to understand the different file systems that support flash, then contact your nearest Intel representative.

Figure 1 illustrates the different file system categories for flash support. Each category may have multiple implementations. For example, in the LFM category, there exist several implementations, each with a different set of features.

The columns in Figure 1 represent the two major types of file system implementations for flash. One uses a linked list approach to file management. The other uses small, equally-sized blocks of memory, sometimes called sectors, to emulate a “drive-like” implementation.

Figure 2 lays out much of the same information as Figure 1, but in a different format. Figure 2 also provides the status of various software file system products Intel is providing/investigating.

Figure 3 provides a file system/manager selection tree. By starting at the top of the page and answering the questions, you can determine the “best fit” file system to meet your needs. Some of these file systems are offered by third party vendors like M-Systems, SystemSoft, and SCM. The rest of the file systems are offered to OEM Intel Flash customers by Intel Corporation as reference code.

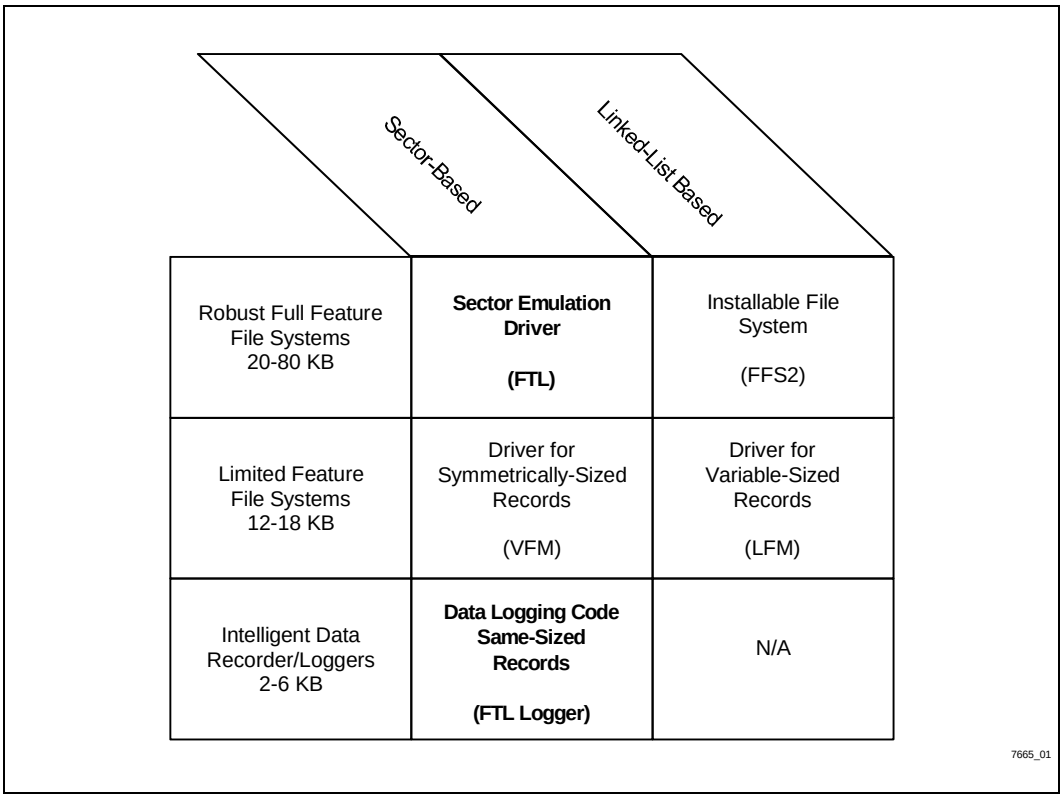


Figure 1. Flash File System/Manager Product Categories

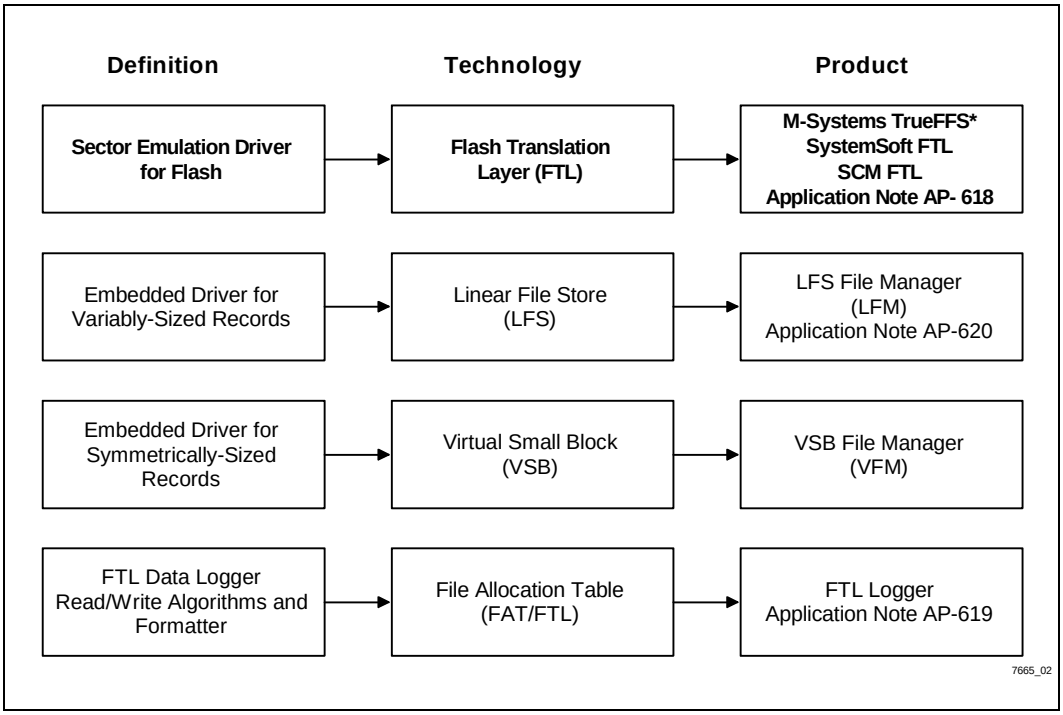
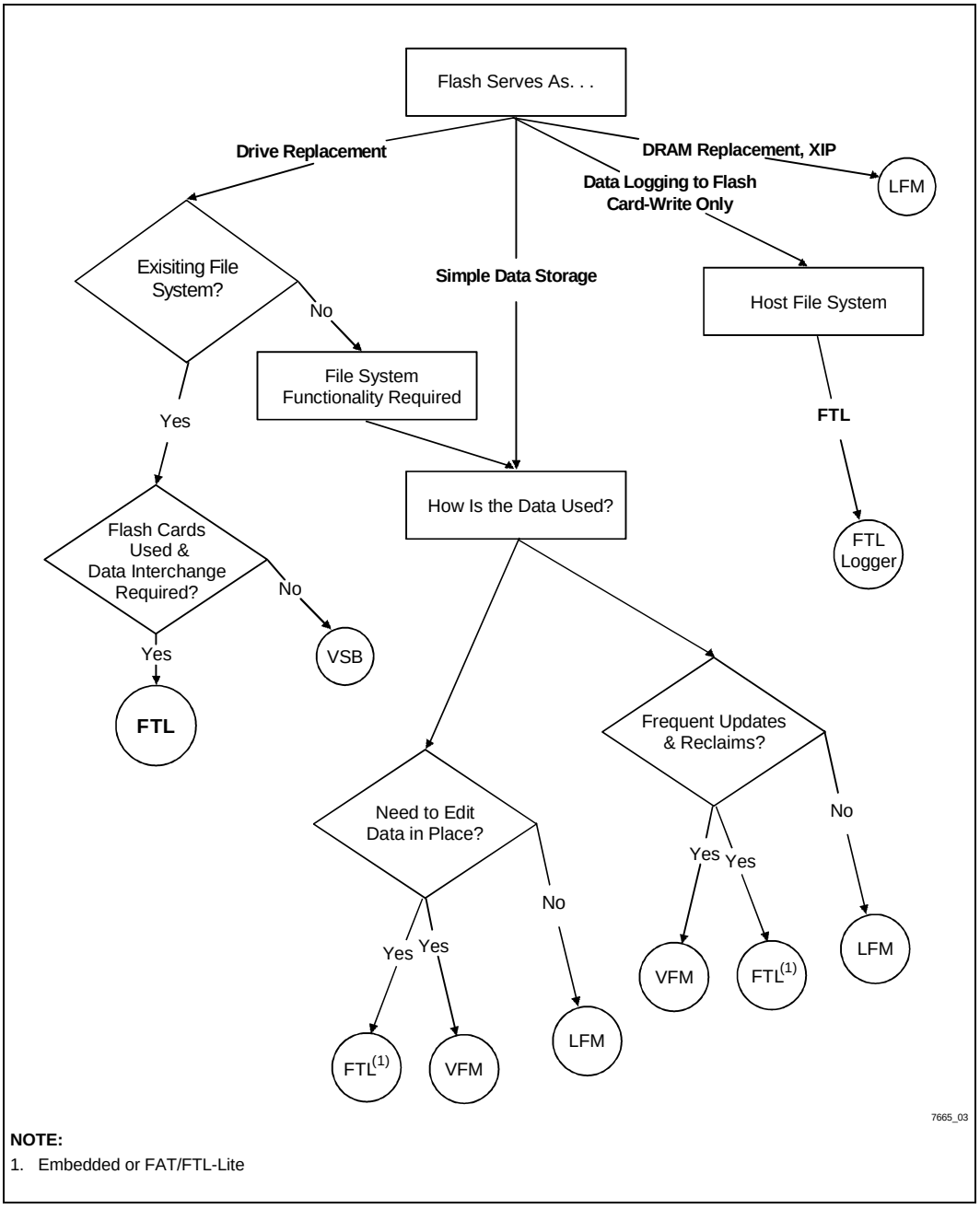


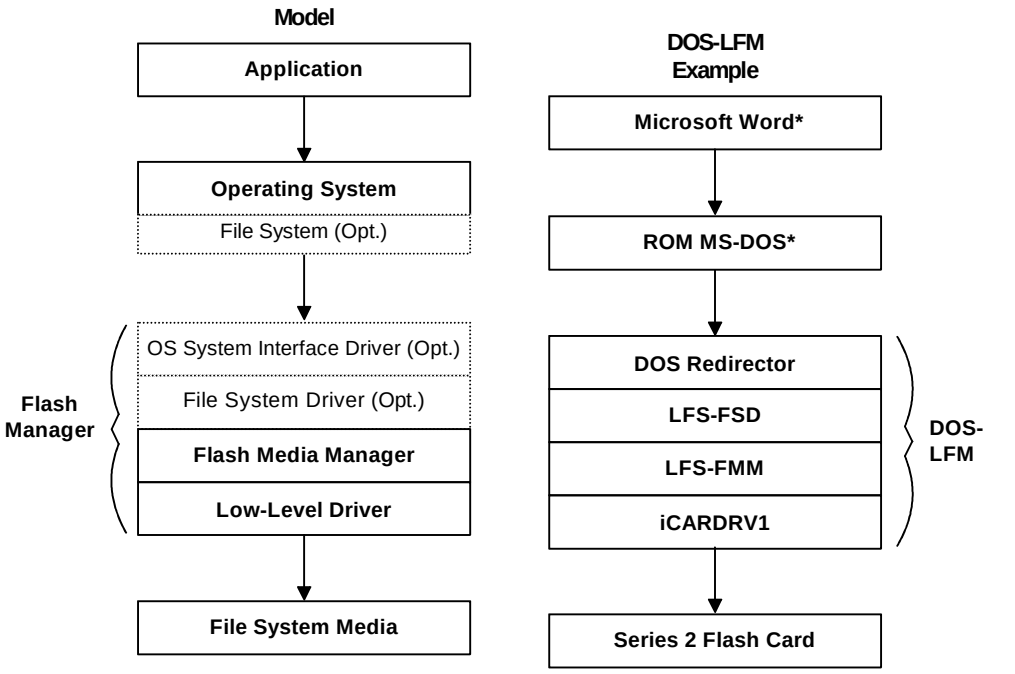
Figure 2. Flash File System/Manager Product Descriptions



NOTE:

1. Embedded or FAT/FTL-Lite

Figure 3. Flash Support Decision Tree



*Other brands and names are the property of their respective owners.

Figure 4. Typical File System Implementation

2.0 FILE SYSTEM DEFINITIONS

2.1 Flash Translation Layer (FTL) Mangers

A Flash Translation Layer (FTL) solution conforms to the PC Card (PCMCIA) Standard FTL specification. This type of flash file system/media manager enjoys wide popularity, with many companies offering FTL solutions. FTL uses an existing sector-based file system to provide the upper level file handling capabilities. In actuality, an FTL driver tricks the upper layer software into believing a normal sector-based drive exists by translating the drive requests which pass through it. Because the FTL solution is used with an existing sector-based operating system, it is relatively small, approximately 24K, thus making FTL a very attractive solution in the DOS world.

The difference between FTL and other flash file systems/media managers, is that FTL provides sectors and, in the case of DOS and DOS-like operating systems, has a File Allocation Table (FAT), rather than linking files and directories, as do some other flash solutions. In terms of overall performance, the best FTL solution on the market performs much better than any other flash media manager. Because FTL emulates a sector-based system, it has several advantages over its counter-parts. FTL has a smaller footprint, fully supports disk recovery utilities, file compression, and it can be cached. Additionally, FTL supports wear leveling and reclamation.

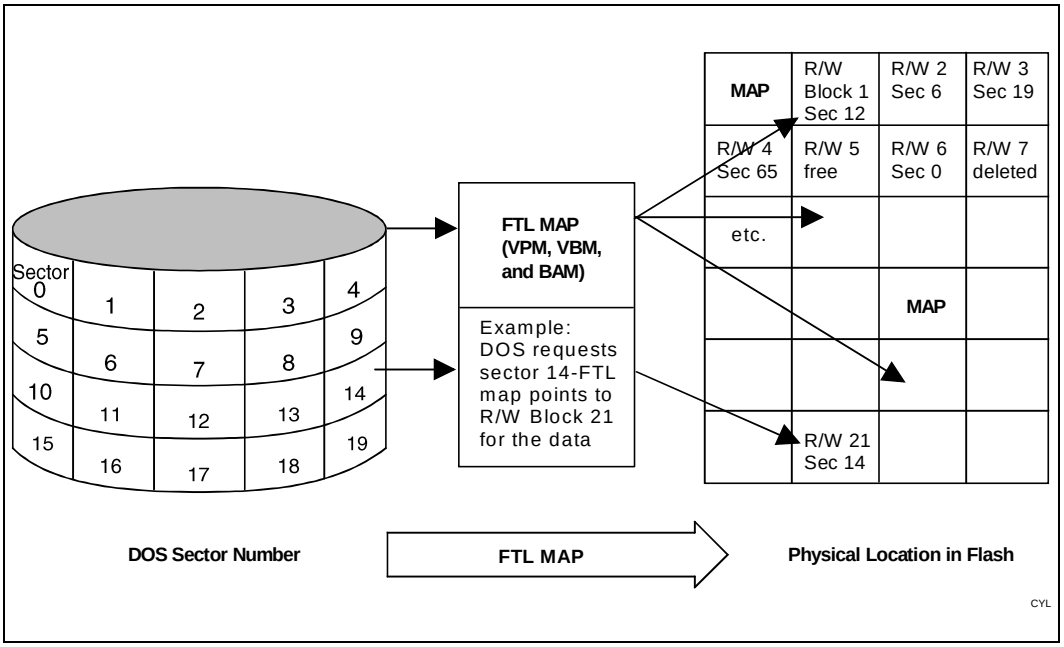


Figure 5. Flash Translation Layer Overview

Figure 6 illustrates a typical FTL structure or format in various flash erase units.

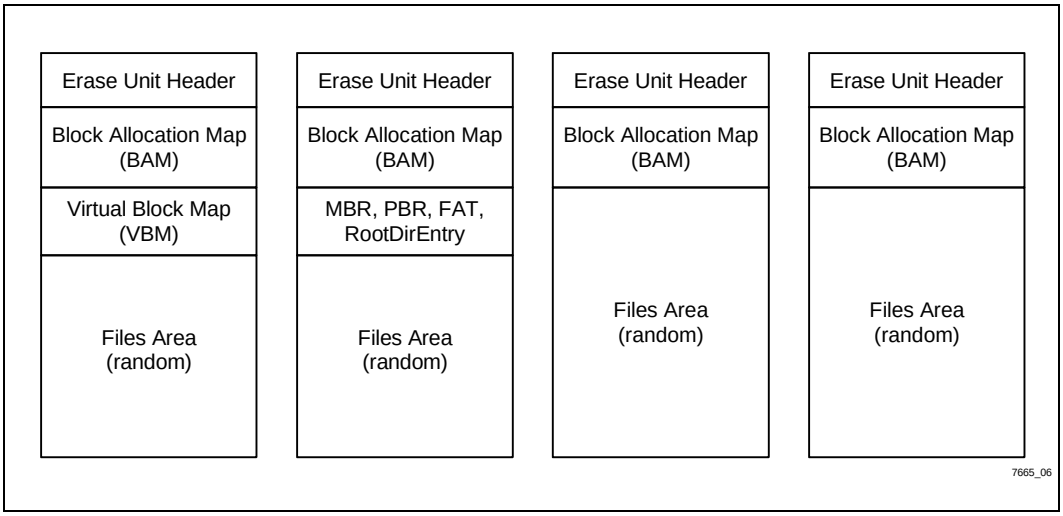


Figure 6. Typical FTL Data Structures

FTL is fast becoming a default feature of PC desktops, notebooks, etc. that have a PC Card capability. Embedded FTL, FAT/FTL-Lite, or Flite* (reduced feature set DOS FAT + FTL), is used in PC consumer peripherals, thus enabling “Back-to-the-PC” capabilty. All of this means “FTL Is IT!” in every sense of the phrase.

Target OS:	Operating Systems with FAT-based or sector-based file systems.
Main purpose:	Provides complete flash media management capabilities, including wear leveling and reclamation.
Related documents:	AP-618 *Software Concerns of Implementing a Resident Flash Disk* (Order 292173).

2.2 Flash File System (FFS2) Managers

A Flash File System (FFS2) in this document refers to a file system that is compliant with Microsoft's specification for their Flash File System Two (FFS2). Microsoft's FFS2 was designed to take into account flash media's special characteristics. The basic concept is that files are stored in a linked-list structure, thus efficiently utilizing the flash storage.

The file system takes into account that flash starts out with all bits set, versus cleared, which is common in a FAT-based system. Because the Flash File System was designed around the properties of flash, the end result is a full feature non-DOS compliant file system. In order to make FFS2 compliant with DOS, it was necessary to design it to be an installable file system by using the DOS network redirector interface. The redirector interface allows the installation of a non-sectored or "alien" file system onto a DOS platform. Therefore, FFS2 installs on a PC similar to the way a network drive is mounted.

Like network drivers, the biggest problem with FFS2 is its size, which stems from it maintaining its file system information on its own, and because of the redirector interface overhead, both take a lot of room. Also, drive utilities cannot be used with FFS2 because of the redirector interface and the non-sector based operation.

FFS2 supports standard file system capabilities, such as creating and deleting files, opening and closing files, editing files, plus wear leveling and reclamation.

Target OS:	Portable to any off-the-shelf operating system.
Main purpose:	Provides complete file systems capabilities, including wear leveling and reclamation.

2.3 LFS File Manager (LFM)

The Linear File Store (LFS) File Manager is Intel's implementation of the PC Card Standard-defined LFS. LFM is a Flash Manager (FM) that provides basic file system functionality for reading and writing variable size file objects. The file objects are stored contiguously in a partition and are arranged in a one-way linked list. The topmost 32 bytes of each file object contain an LFS header. The header contains basic file information including a link to the next file object, which begins immediately after the file.

LFM was designed and implemented by Intel specifically for the embedded market, where a certain amount of functionality may be sacrificed in order to obtain a file system for flash with a small footprint. LFM uses PC Card defined LFS structures to create a link-list type of file system. Each file is guaranteed to be contiguously laid out in order to support eXecute-In-Place (XIP) applications, which are primarily found in embedded systems that need to use memory efficiently in order to keep costs low.

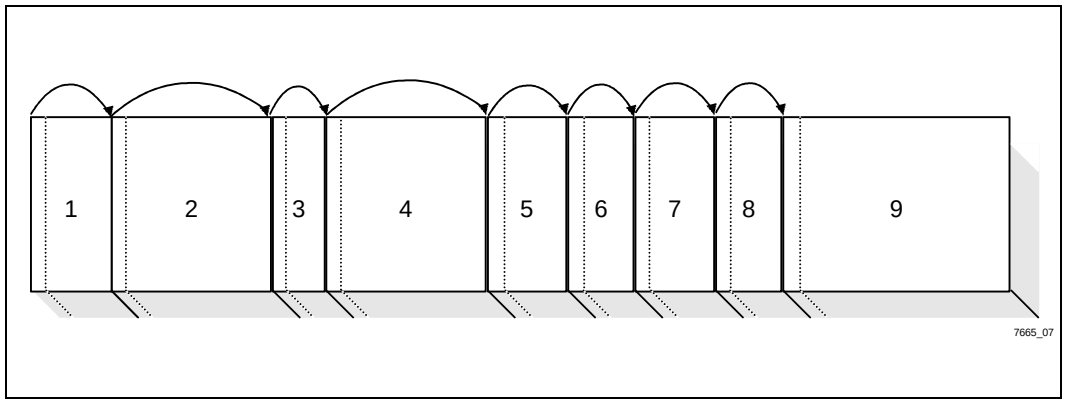


Figure 7. LFM File Links

LFM allows multiple partitions, with multiple files open for reading and one file per partition open for writing. The LFM reference code is well-defined, with a clear API for the low-level hardware driver as well as for the high-level application. One advantage LFM has over many other file system options is that it is operating system independent and is written in "C" to be portable. Intel provides the LFM code to our OEM flash customers as a reference design.

Target OS:	Portable to any off-the-shelf operating system.
Main purpose:	Provides limited file systems capabilities, includes file reclamation.
Related documents:	AP-620 *LFS File Manager Software: LFM* (Order 292175)

2.4 VSB File Manager (VFM)

Virtual Small Block (VSB) File Manager is another mid-range sector-based flash manager, similar to its FAT+FTL big brother. The purpose for VFM is to provide a low-cost embedded Resident Flash Array (RFA) implementation without requiring DOS compatibility or PC Card socket services.

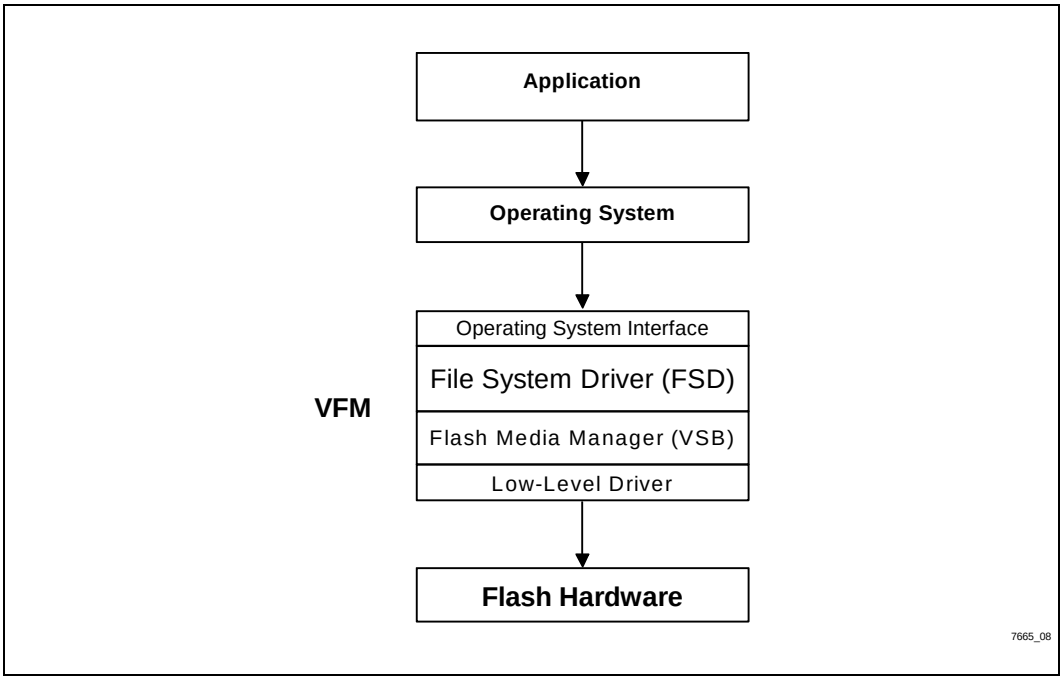


Figure 8. VFM Software Layers

Like an FTL solution, VFM is sector-based, emulating small block characteristics of a hard disk, even when used with large block flash media. Each block contains a VSB Allocation Table (VAT) at the beginning of each block. The VAT is used to determine the logical identity of each Virtual Small Block. Intel's provides VFM software to our OEM flash customers as a reference design.

Target OS:	Small embedded operating systems and applications.
Main purpose:	Provides complete basic file systems capabilities, including reclamation for applications where the data primarily stays on the embedded system. For applications where data needs to be transported back to the PC on the flash media, FTL is the recommended solution.

2.5 FTL Logger

The FTL Logger reference source code provides the embedded application the ability and knowledge to log or record information to a flash PC Card. The format is compatible with any Flash Translation Layer enabled desktop/notebook with a PC Card slot, without requiring an FTL file system to be installed on the embedded system. This approach is an excellent way to improve write speed to flash by eliminating file system overhead, which is particularly important in an embedded system running at a slow clock speed. This approach is also very code size conservative, as code-intensive file system format knowledge is not required. Once the data is stored, the flash PC Card may then be inserted into a host system supporting an FTL solution and the files read, edited, copied, etc.

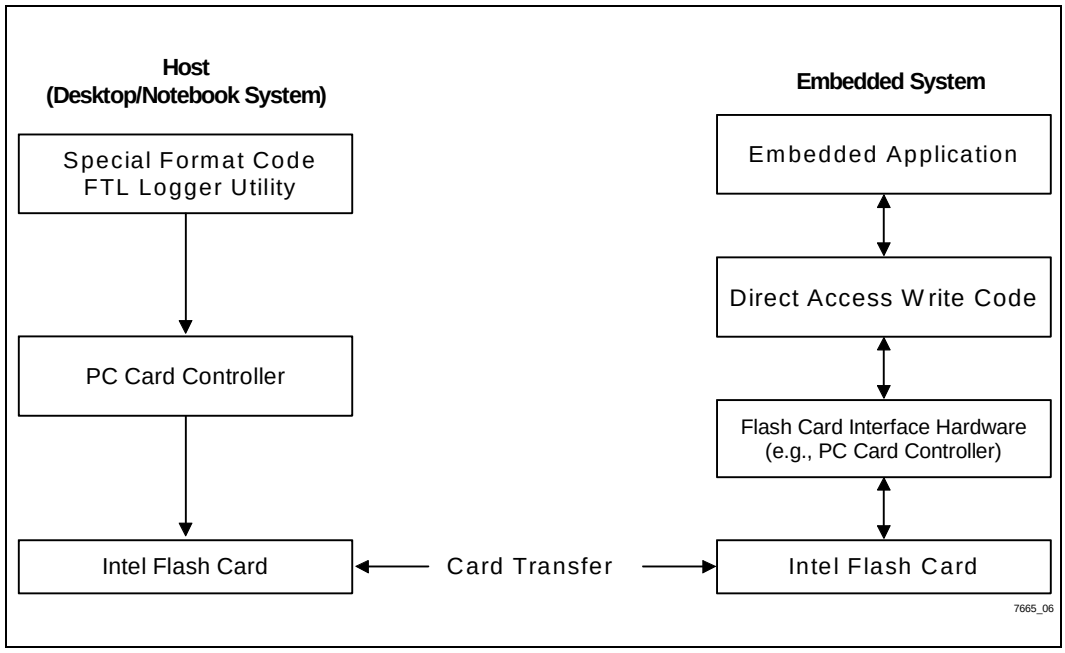


Figure 9. Typical FTL Logger Implementation

Intel provides FTL Logger source code to our flash customers as a reference design. It is available from the Intel ASMO BBS and from Intel field sales representatives.

Target OS:	Used by small embedded operating systems or applications.
Main purpose:	Data logging or recording on to a pre-formatted flash PC Card card in a FTL format.
Related documents:	AP-619 *FTL Logger Exchanging Data with FTL Systems* (Order 292174)

APPENDIX A
DEFINITIONS AND ACRONYMS

FAT	File Allocation Table.
FSD	File System Driver, the core driver needed to create a complete File System.
FTL	Flash Translation Layer, the core driver needed to create a complete File System.
IFS	Installable File System.
LFS	Linear File Store (as defined by PC Card).
LFS extended header	Additional descriptive data (not defined by PC Card) that exists between the LFS header and the file object's data, but contributes to the overall size of the file object.
LFS header	A PC Card-defined 32-byte field describing a quantity of data stored in an LFS partition.
LFM	LFS File Manager.
Media	A storage mechanism made from flash memory components; an RFA or a flash card.
Partition	An integral number of contiguous flash erase blocks.
PC Card	The standard which defines the small "PCMCIA" form factor and software.
PCMCIA	Personal Computer Memory Card International Association.
Reclaim	The process by which unused flash media is made available for use.
RFA	Resident Flash Array, a fixed array of flash memory components within an embedded computer.
VAT	VSB Allocation Table.
VFM	VSB File Manager.
VSB	Virtual Small Block.

APPENDIX B
ADDITIONAL INFORMATION

Related Intel Documents (1,2)

Order Number	Document
292173	AP-618 *Software Concerns of Implementing a Resident Flash Disk*
292174	AP-619 *FTL Logger Exchanging Data with FTL Systems*
292175	AP-620 *LFS File Manager Software: LFM*

NOTES:
1. Please call the Intel Literature Center at (800) 548-4725 to request Intel documentation. International customers should contact their local Intel or distribution sales office.
2. Visit Intel's World Wide Web home page at http://www.Intel.com for technical documentation and tools.