Flash / SmartMedia™ File System

2000. 02 .17

Memory Product &
Technology Division

Flash File System Introduction

- Flash file system,or FTL(Flash Translation Layer), is to make flash memory appear to the system like a disk drive.
- Why is FTL needed?
  - Flash are not 100% perfect . It needs bad block management.
  - Flash is erased in blocks(typical 16KB) larger than disk sectors(512B)
  - Flash has a limited number of erase cycles(1M Cycles) So it needs wear-leveling algorithm.
  - Flash is essentially non-writable (must be erased before it can be written)

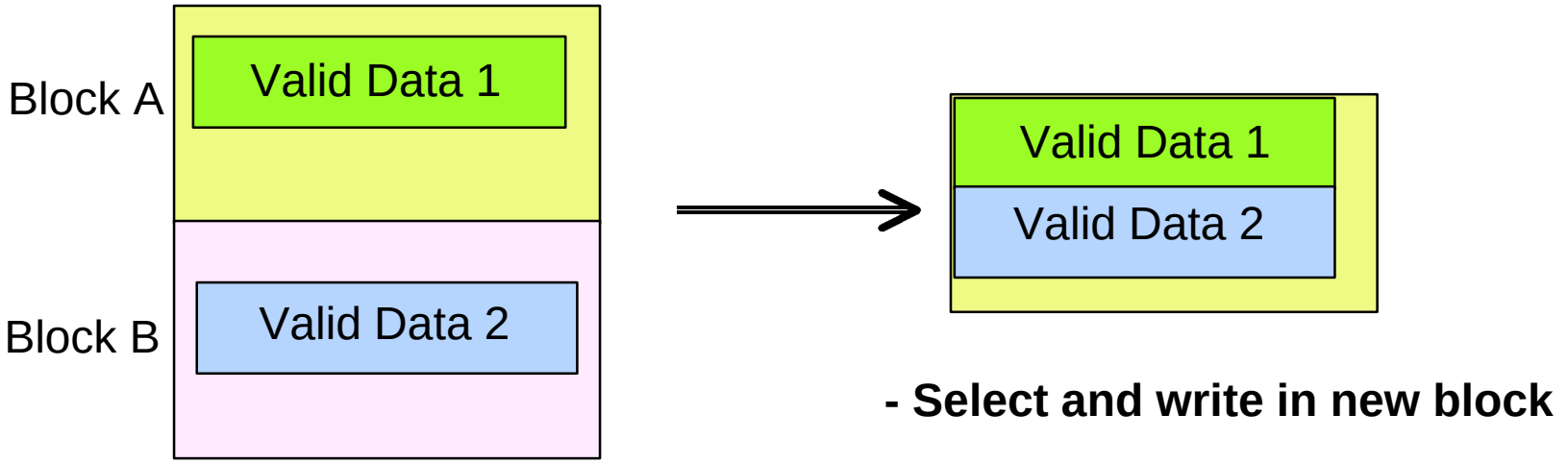
Algorithms for Flash File System(1)

- **Wear-leveling algorithm**

  - Monitoring each block writing numbers or changing each block for writing.

  => for endurance of flash memory

- **Garbage collection algorithm**



- Select and write in new block

- Erase previous block

Algorithms for Flash File System(2)

- **Flash updating algorithm**

  - Updating into empty blocks reduces memory demands and avoid non- rewritable feature.

- **Background erase algorithm**

- Background erase algorithm is erasing all erasable block while idle time.

(erasable block is to be alternated block for Flash updating algorithm.)

Examples of Flash File System

- **TrueFFS& Flite(M-Systems)**

  - TrueFFS uses block allocation map for native file system and FTL.

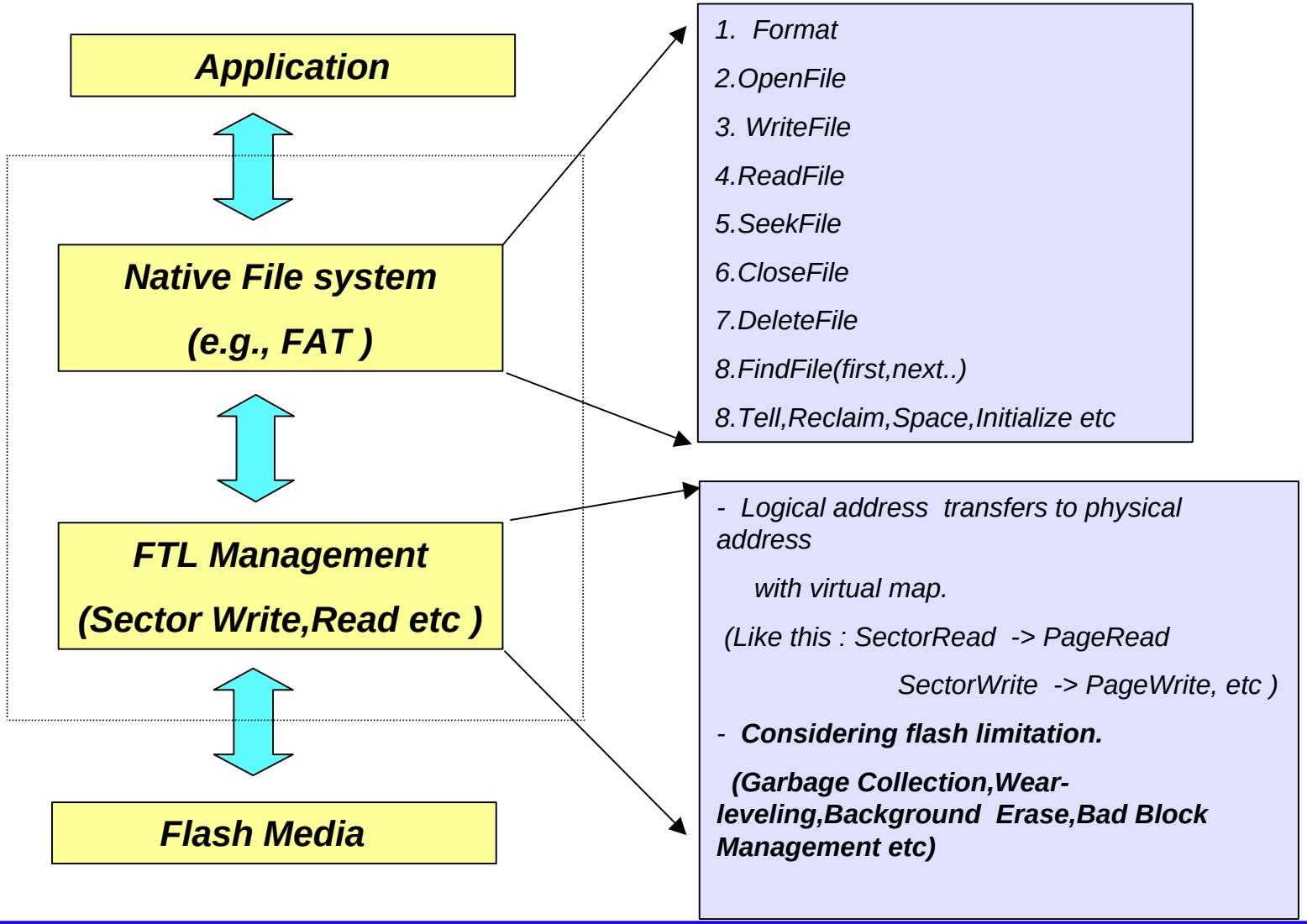
  - FLite includes FTL for embedded Flash disk.

- **IVFM(Intel Virtual Small Block File System)**

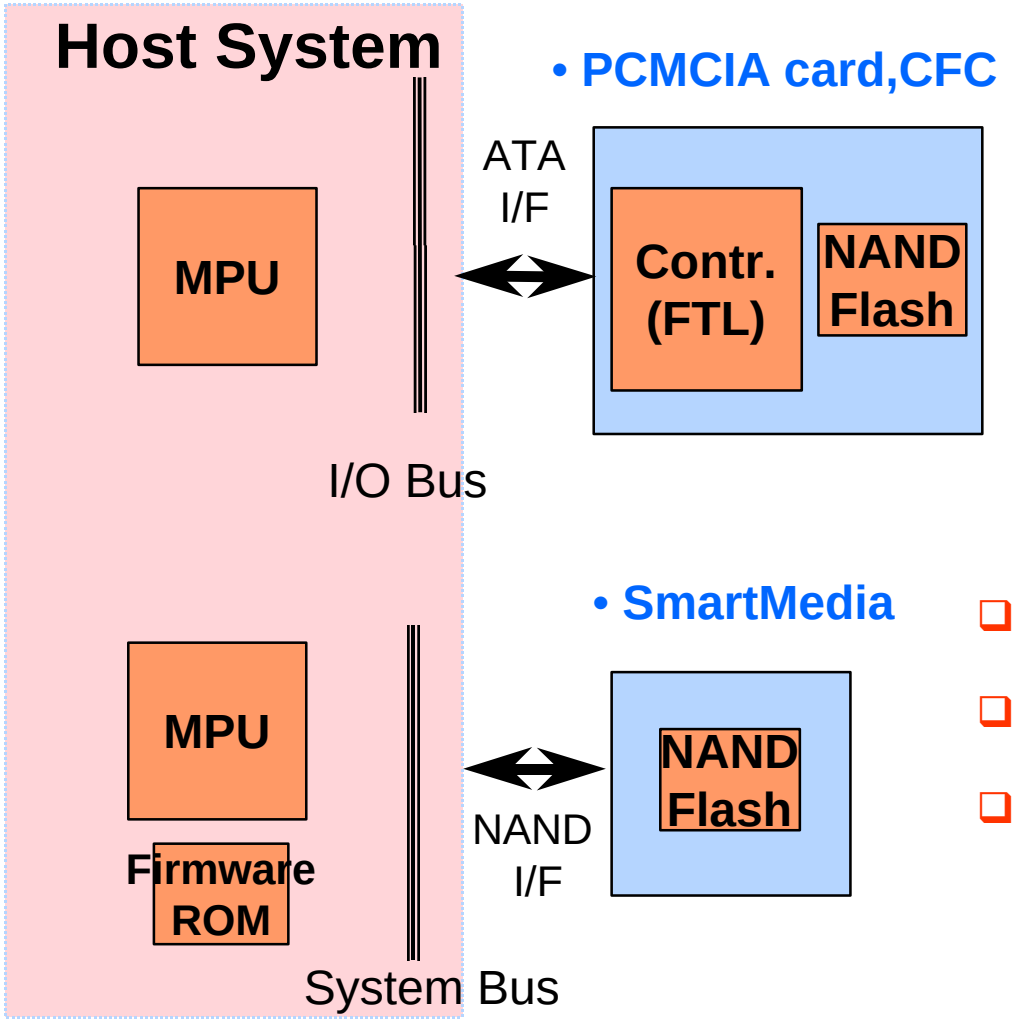
  - Recommended for non-removable embedded file/data storage applications

  Examples : Printer,Fax,Scanner,Digital answering machines

Main API required by System



FTL & Flash Card

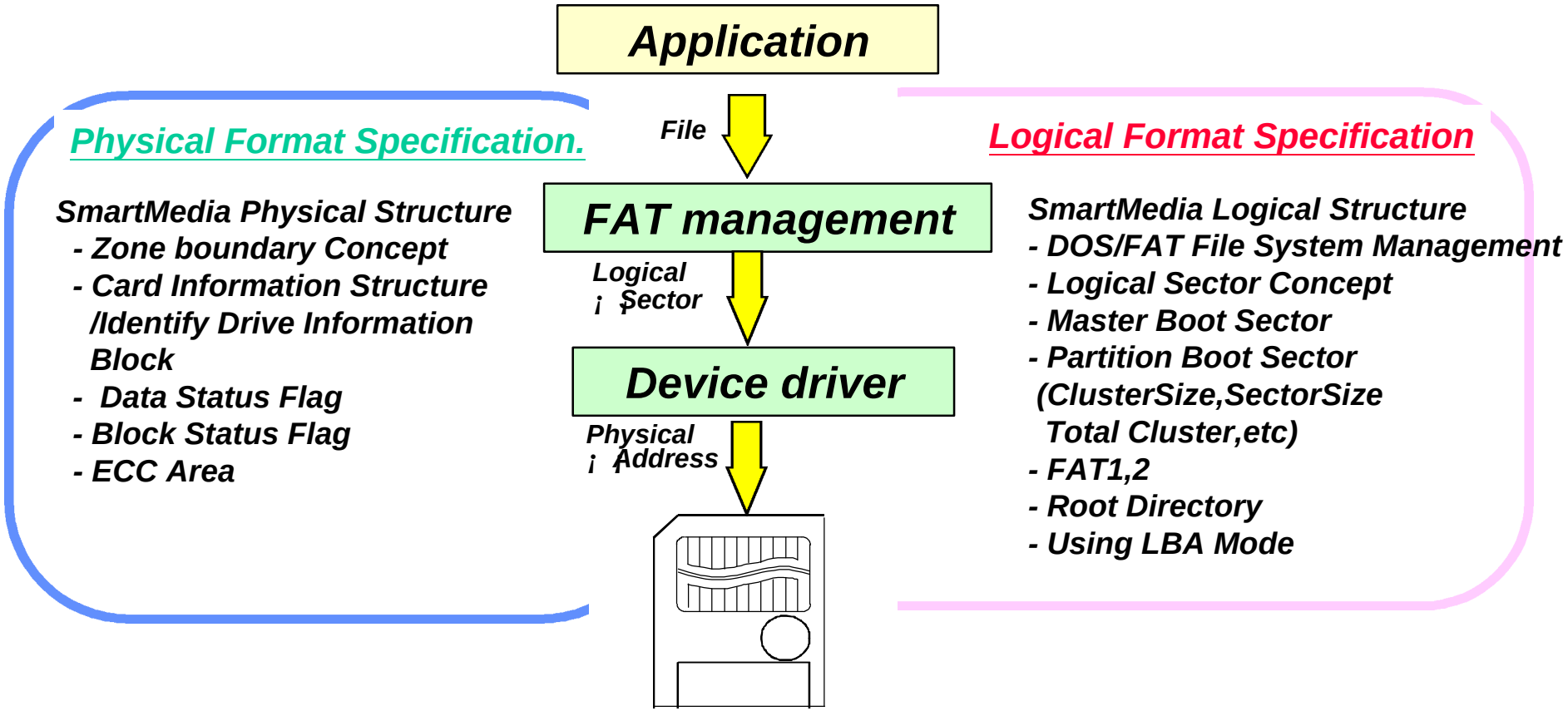


- High overhead cost for card assembly
- Flash file management by controller-proprietary firmware

- No overhead cost for assembly
- Direct access by host system bus
- Flash file management by host firmware needs to be standardized for compatibility

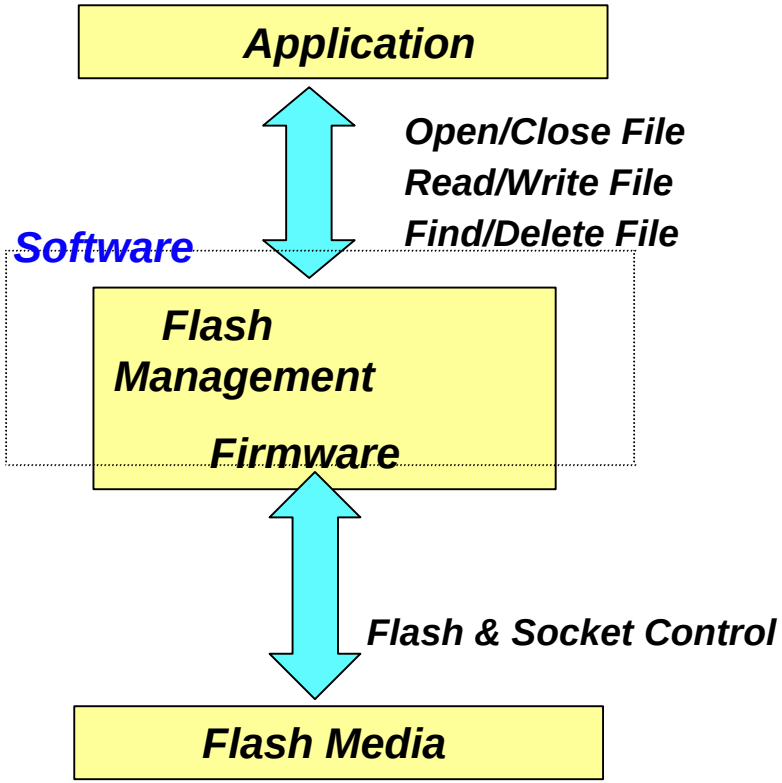
File System for SmartMedia

- SmartMedia Forum formulated specification for file system
- Forum already standardized up to 128MB SmartMedia.

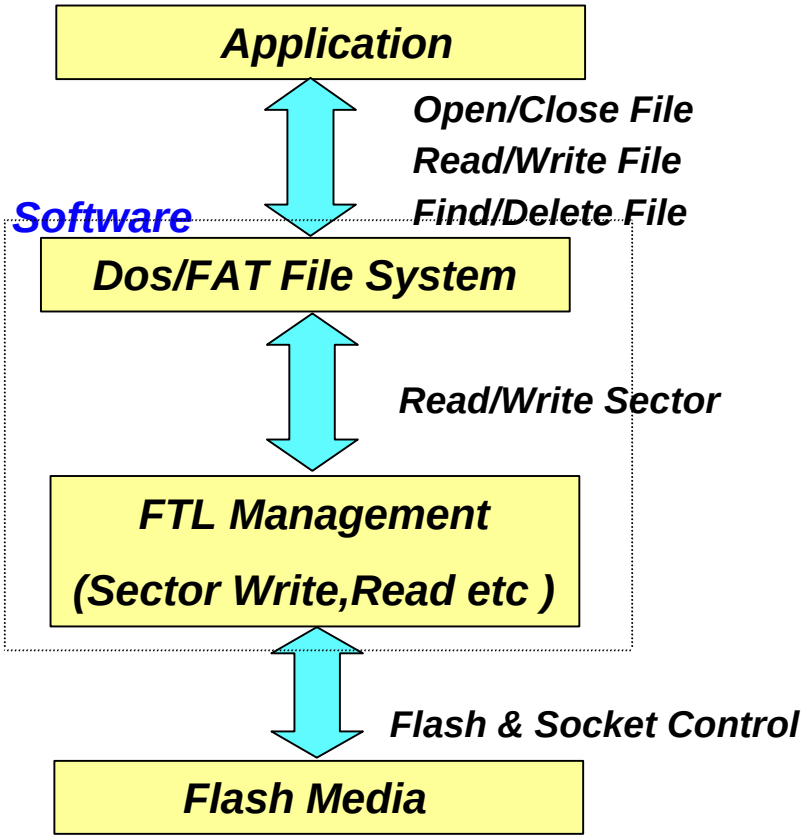


Embedded Flash & SmartMedia

- In Case of embedded Flash



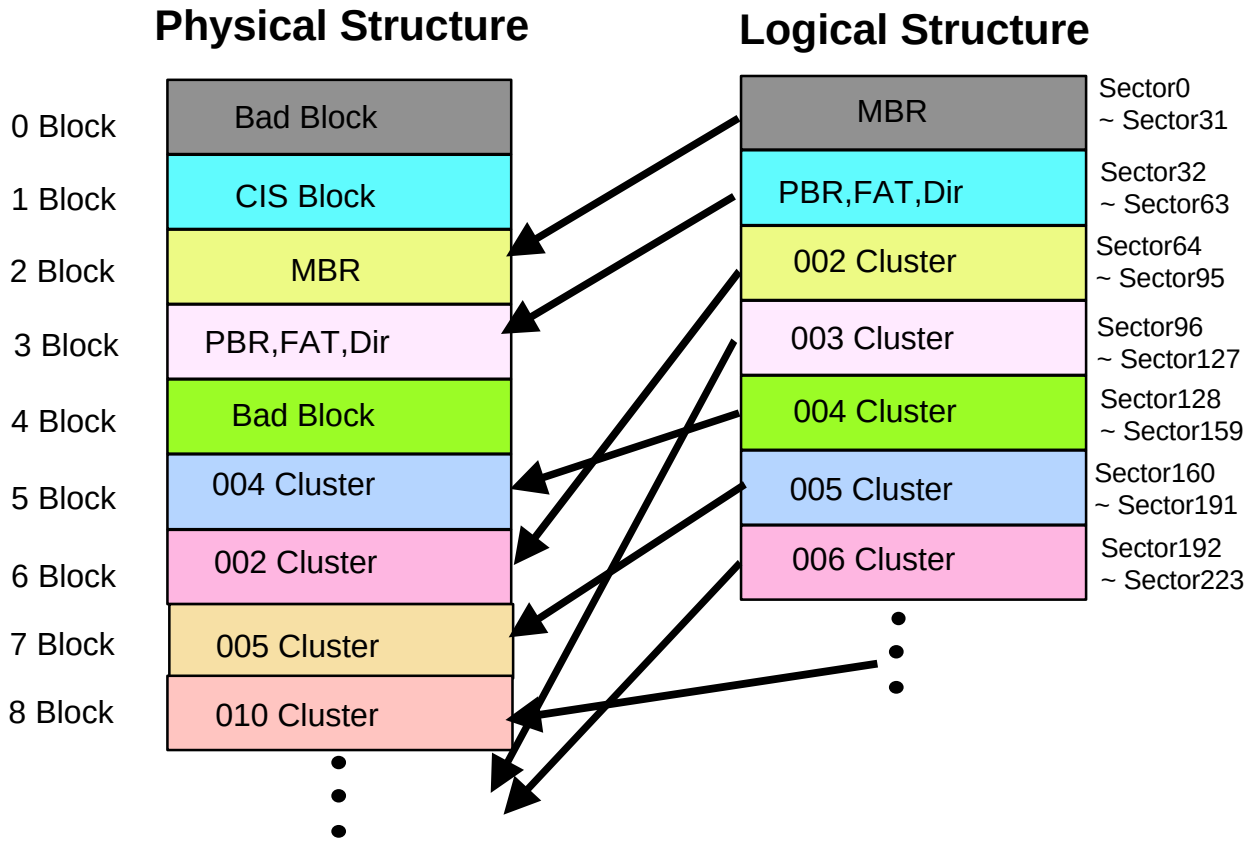
- In Case of SmartMedia



** SmartMedia requirement is just DOS/FAT file system*
,but embedded Flash requirement is various

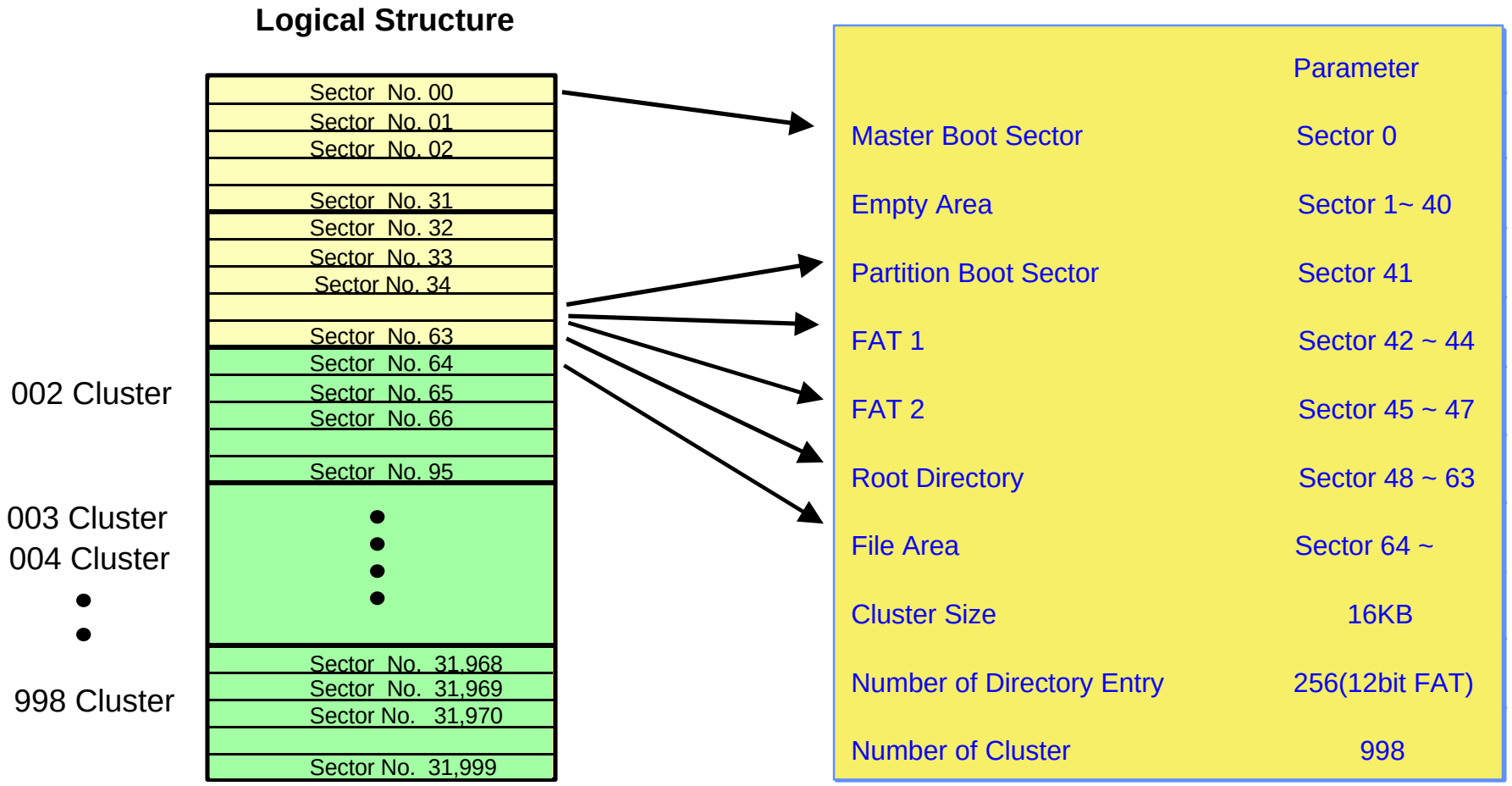
How Logical/Physical Structures are interrelated In Smartmedia

Irrelevant to physical address, logical structures are pre-defined and gives flexibility in the memory usage



What is Logical Format in SmartMedia?

In case of 16MB SmartMedia



What is Physical Format in SmartMedia?

2MByte

NAND Structure
264Bytex16Pages x 512Blocks

Block No.000 (CIS)
Block No.001
Block No.002
Block No.003
⋮
Block No.510
Block No.511

Block Structure
1Block=16Page

Page No.00
Page No.01
Page No.02
Page No.03
⋮
Page No.14
Page No.15

Sector Data Structure
1Sector=2Pages

	Even Page	Odd Page
0-255	Data Area-1	Data Area-2
256	Reserved Area	ECC Area-2
257		
258		
259		Block Address-2
260	Data Status Flag	
261	Block Status Flag	ECC Area-1
262	Block Address-1	
263		

4MByte
8MByte
Higher Densities

NAND Structure
528Bytex16,32Pages x 512Blocks

Block No.000 (CIS)
Block No.001
Block No.002
Block No.003
⋮
Block No.510
Block No.511
⋮
Block No.1022
Block No.1023

Block Structure
1Block=16,32Page

Page No.00
Page No.01
Page No.02
Page No.03
⋮
Page No.14
Page No.15

Sector Data Structure
1Sector=1Page

0-255	Data Area-1
256-511	Data Area-2
512-515	Reserved Area
516	Data Status Flag
517	Block Status Flag
518-519	Block Address-1
520-522	ECC Area-2
523-524	Block Address-2
525-527	ECC Area-1

SEC's Solution for Flash & SmartMedia

• **Hardware : PC interfaced board for operating NAND Flash.**

- **ISA Demo Board**
- **EPP Mode Demo Board (Available now)**

• **Software**

- **Sample code for basic operation.**
- **Reference source for embedded Flash file system.**

  - Under development

- **Reference source for SmartMedia file system**

  - Plain structured SmartMedia file system

  - Layer structured SmartMedia file system (Available now)

  (full version, Light platform version)

File System Reference I for SmartMedia(I)

PC Utility for SmartMedia File Management

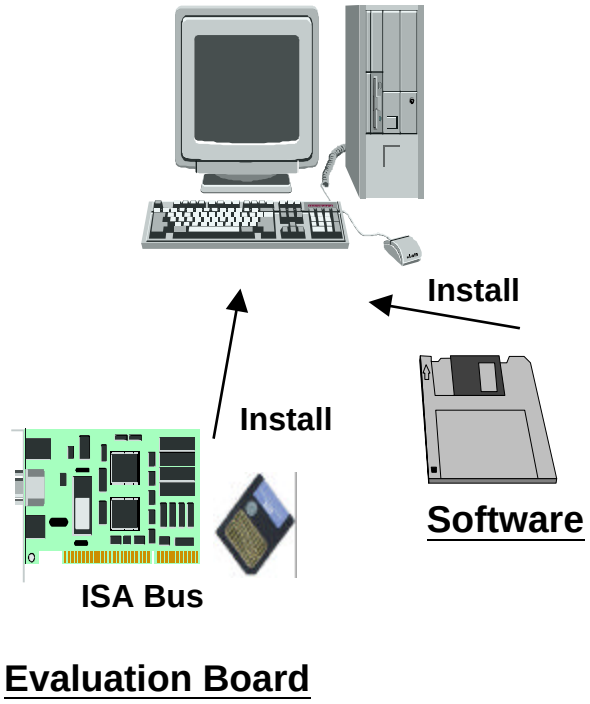
- Complied utility running under DOS with a proprietary card in ISA or parallel port
- C source code may be referenced for SmartMedia file system
- Built-in “SmartMedia Optimized Format” up to 128MB
- Built-in software/hardware ECC generation
- Footprint

  ROM : 130KByte(Including sample application)

  Literal file system below 50KB

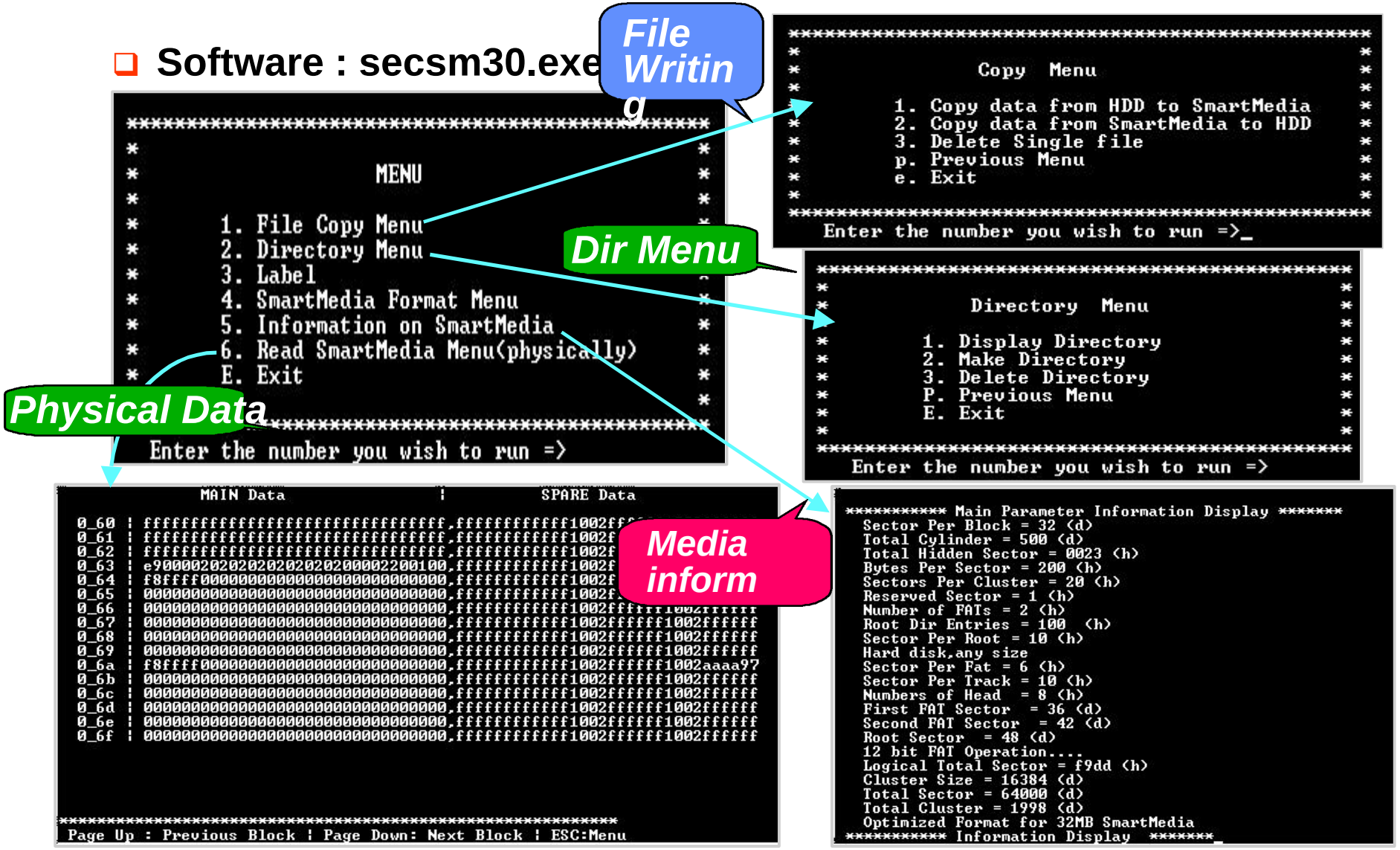
  RAM : 12KBytes
- Detailed manual available
- Supporting functions
  - Write/Read/Delete file
  - Display/Make/Delete sub-directory
  - Volume labeling
  - Analyzing capability

  (Information of physical block data and FAT)



File System Reference I for SmartMedia(II)

- Software : secsm30.exe



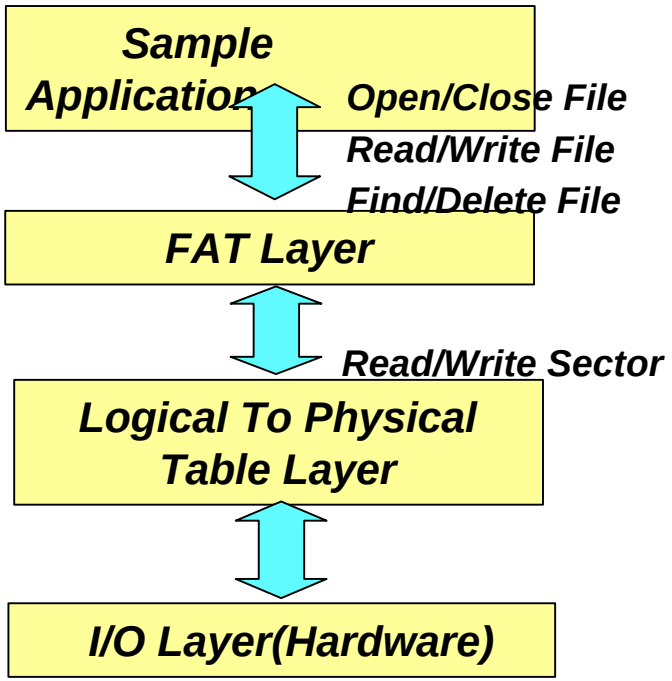
File System Reference II for SmartMedia

- **Layer structured SmartMedia file system API (full version)**

- Comes with C source code and supports ISA ,Parallel Port
- Built-in “SmartMedia Optimized Format” up to 128MB
- Built-in software/hardware ECC generation
- 25 API Supported
- Detailed manual available
- Footprint

  ROM size : 73KB ,RAM size: 35KB
- Supporting main API
  - Open/Close/Read/Write/Find File ...
  - Display/Make/Delete Sub-Directory ..

-- Layer Structure --



File System Reference III for SmartMedia

- **Layer structured SmartMedia file system API (Light version)**

  – **Same API support as full version**
  – **Optimized for "minimal footprint"**
  – **Detailed manual available**
  – **Footprint**
  ROM Size : 58KB ,RAM Size: 2.5KB
  – **Using one or two block as bad block for physical to logical block table buffer**