# The Fast File System

M.K. McKusick, W. Joy, S.L. Leffler, R.S. Fabry, "A Fast File System for UNIX", ACM Transactions on Computer Systems, Vol. 2, No. 3, Aug. 1984

- Recall: Limitations of Original UNIX File System
- Fast File System (FFS)
  - Increase Block Size
  - Make File System "disk-aware"
  - Some Functional Improvements

# Recall: Problems with Original Unix File System

- Original file system uses 512/1024-byte blocks.

- Inodes kept separately from data, causing long seeks to access data.
- Inodes of files in a common directory not kept together, causing low performance when searching directories.
- Data blocks of a file are not stored together.
- Free list quickly scrambles, increasing overhead of finding free blocks.

Original file system treats disk as Random Access Device!

## "Fast FS" (FFS, ca. 1984)

M.K. McKusick, W. Joy, S.L. Leffler, R.S. Fabry, "A Fast File System for UNIX", ACM Transactions on Computer Systems, Vol. 2, No. 3, Aug. 1984

Three-pronged approach:

1. Increase block size
2. Make file system disk-aware
3. Functional improvements

## FFS: Increase Block Size

Increase block size from 512 byte to at least 4096 byte.

- Block size power-of-two multiple of 4096 byte
- Defined during file system creation
- Stored as parameter in superblock

# FFS: Block Size and Fragmentation

Observation: Typical UNIX files are small compared to large blocks.

M.K. McKusick, *et al.*

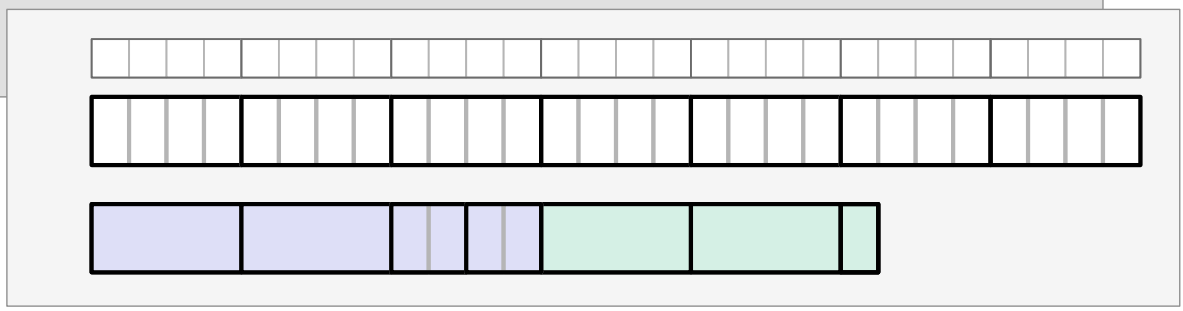
Table I. Amount of Wasted Space as a Function of Block Size

| Space used (Mbytes) | % Waste | Organization |
|---|---|---|
| 775.2 | 0.0 | Data only, no separation between files |
| 807.8 | 4.2 | Data only, each file starts on 512-byte boundary |
| 828.7 | 6.9 | Data + inodes, 512-byte block UNIX file system |
| 866.5 | 11.8 | Data + inodes, 1024-byte block UNIX file system |
| 948.5 | 22.4 | Data + inodes, 2048-byte block UNIX file system |
| 1128.3 | 45.6 | Data + inodes, 4096-byte block UNIX file system |

# Blocks vs. Fragments

Solution: Split blocks into **Fragments**

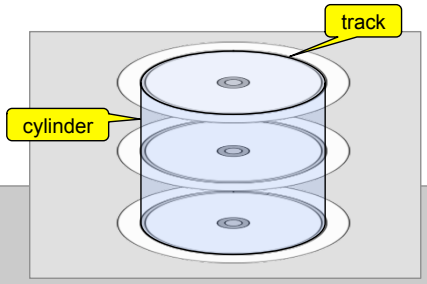
- Fragment size defined at file system creation time
- Example: 4096/1024 has 4kB blocks and 1kB fragments.



# FFS Organization: Making FS Disk-Aware

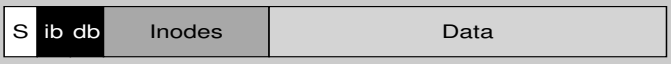
1. **Cylinder Groups**
2. File System **Parameterization**

# FFS Organization: Cylinder Groups




**Cylinder Groups**

- groups of multiple adjacent disk cylinders.
- each group maintains own copy of superblock, inode bitmap, data bitmap, inodes, and data blocks:




Allocation of directories and files:

- "keep related stuff together"
- inodes and data blocks
- blocks of same file
- files and directories

# FFS: Minimizing Rotational Latency

## File System **Parameterization**

Goal: Parameterize **processor capabilities** and **disk characteristics** so that blocks can be **allocated** in an optimal, configuration-dependent way.

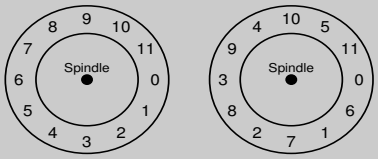
Allocate new block **rotationally well-positioned**.

Disk Parameters:

- number of blocks per **track**
- disk **spin rate**.

CPU Parameters:

- expected time to **service interrupt** and **schedule new disk transfer**




# FFS: Performance

Table IIa. Reading Rates of the Old and New UNIX File Systems

| Type of file system | Processor and bus measured | Speed (Kbytes/s) | Read bandwidth % | % CPU |
|---|---|---|---|---|
| Old 1024 | 750/UNIBUS | 29 | 29/983 3 | 11 |
| New 4096/1024 | 750/UNIBUS | 221 | 221/983 22 | 43 |
| New 8192/1024 | 750/UNIBUS | 233 | 233/983 24 | 29 |
| New 4096/1024 | 750/MASSBUS | 466 | 466/983 47 | 73 |
| New 8192/1024 | 750/MASSBUS | 466 | 466/983 47 | 54 |

Table IIb. Writing Rates of the Old and New UNIX File Systems

| Type of file system | Processor and bus measured | Speed (Kbytes/s) | Write bandwidth % | % CPU |
|---|---|---|---|---|
| Old 1024 | 750/UNIBUS | 48 | 48/983 5 | 29 |
| New 4096/1024 | 750/UNIBUS | 142 | 142/983 14 | 43 |
| New 8192/1024 | 750/UNIBUS | 215 | 215/983 22 | 46 |
| New 4096/1024 | 750/MASSBUS | 323 | 323/983 33 | 94 |
| New 8192/1024 | 750/MASSBUS | 466 | 466/983 47 | 95 |

M.K. McKusick, *et al.*

## FFS: Functional Enhancements

1. Arbitrary-length File Names
2. File Locking
3. Symbolic Links
4. “Rename” system call added
5. Quotas

## The Fast File System

- Recall: Limitations of Original UNIX File System
- Fast File System (FFS)
  - Increase Block Size
  - Make File System “disk-aware”
  - Some Functional Improvements