

Disk Hardware

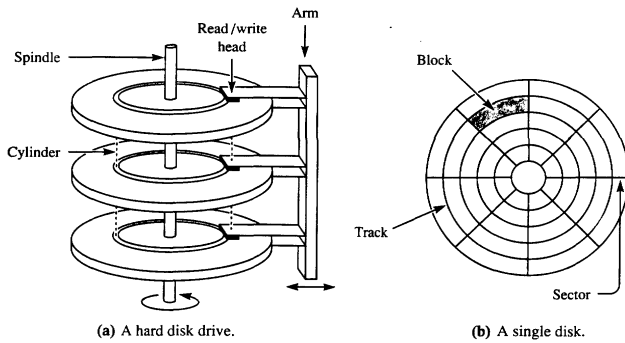


Diagram from *Computer Science*, Volume 2, J. Stanley Warford, Heath, 1991.

- Arm can move in and out
 - Read / write head can access a ring of data as the disk rotates
- Disk consists of one or more *platters*
 - Each platter is divided into rings of data, called *tracks*, and each track is divided into *sectors*
 - One particular platter, track, and sector is called a *block*

1

Fall 2001, Lecture 31

Disk Hardware (cont.)

- Typical disk today (Compaq 40GB Ultra ATA 100 7200RPM hard disk = \$369):
 - 16383 cylinders, 16 heads, 63 sectors/track
 - 16 platters * 16383 tracks/platter * 63 sectors/track * 4048 bytes/sector * 1/1024³ GB/byte = 63GB unformatted
 - 7200 rpm spindle speed, 8 ms average seek time, 100 MBps data transfer rate
- Trends in disk technology
 - Disks get smaller, for similar capacity
 - Faster data transfer, lighter weight
 - Disk are storing data more densely
 - Faster data transfer
 - Density improving faster than mechanical limitations (seek time, rotational delay)
 - Disks are getting cheaper (factor of 2 per year since 1991)

2

Fall 2001, Lecture 31

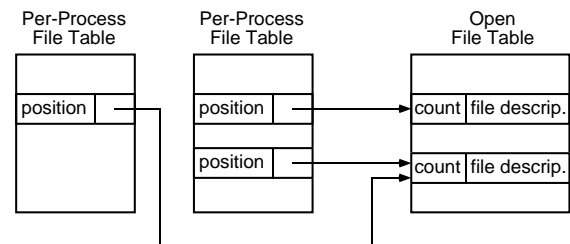
Data Structures for Files

- Every file is described by a *file descriptor*, which may contain (varies with OS):
 - Type
 - Size
 - Access times — when created, last accessed, last modified
 - Owner, group
 - Access permissions — read, write, etc.
 - Link count — number of directories that contain this file
 - Blocks where file is located on disk
- Not included:
 - Name of file

3

Fall 2001, Lecture 31

OS Data Structures for Files

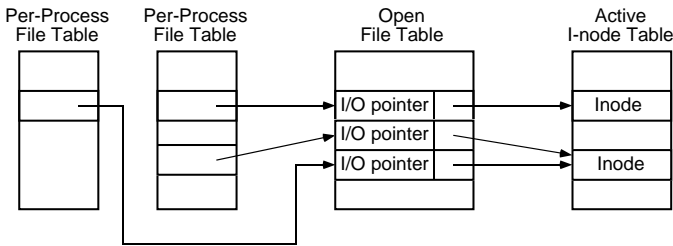


- *Open file table* (one, belongs to OS)
 - Lists all open files
 - Each entry contains:
 - A *file descriptor*
 - Open count — number of processes that have the file open
- *Per-process file table* (many)
 - List all open files for that process
 - Each entry contains:
 - Pointer to entry in open file table
 - Current position (offset) in file

4

Fall 2001, Lecture 31

UNIX Data Structures for Files



- **Active Inode table** (one, belongs to OS)
 - Lists all active *inodes* (file descriptors)
- **Open file table** (one, belongs to OS)
 - Each entry contains:
 - Pointer to entry in active inode table
 - Current position (offset) in file
- **Per-process file table** (many)
 - Each entry contains:
 - Pointer to entry in open file table

5

Fall 2001, Lecture 31

Disk Data Structures for Files

- The file descriptor information must also be stored on the disk, for persistence
 - Includes all the basic information listed on the previous slides
 - All *inodes* (file descriptors) are stored in a fixed-size array on the disk called the *ilist*
 - The size of the ilist array is determined when the disk is initialized
 - The index of a file descriptor in the array is called its *inode number*, or *inumber*
- File descriptors are stored:
 - Originally, together on the inner (or outer) track
 - Then, together on the middle track (why?)
 - Now: there are small file descriptor are spread out across the disk, so as to be closer to the file data

6

Fall 2001, Lecture 31

UNIX File System

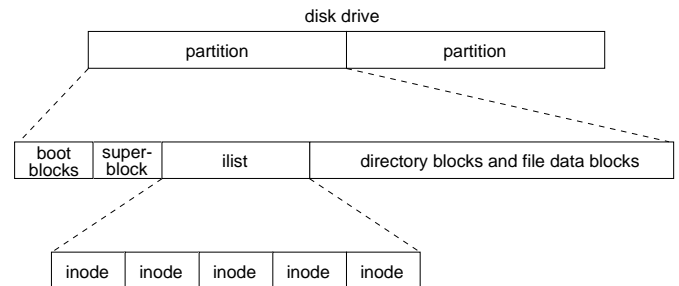
- A file descriptor (*inode*) represents a file
 - All *inodes* are stored on the disk in a fixed-size array called the *ilist*
 - The size of the ilist array is determined when the disk is initialized
 - The index of a file descriptor in the array is called its *inode number*, or *inumber*
 - Inodes for active files are also cached in memory in the *active inode table*
- A UNIX disk may be divided into *partitions*, each of which contains:
 - Blocks for storing directories and files
 - Blocks for storing the ilist
 - Inodes corresponding to files
 - Some special inodes
 - Boot block — code for booting the system
 - Super block — size of disk, number of free blocks, list of free blocks, size of ilist, number of free inodes in ilist, etc.

7

Fall 2001, Lecture 31

UNIX File System (cont.)

- High-level view:



- Low-level view:

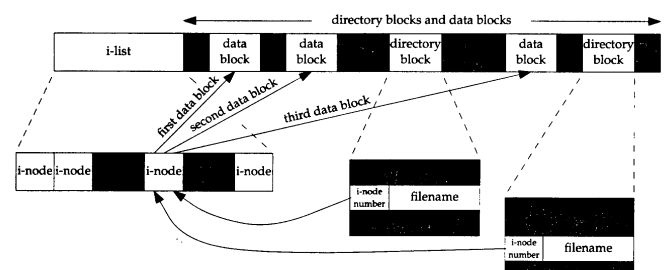


Diagram from *Advanced Programming in the UNIX Environment*, W. Richard Stevens, Addison Wesley, 1992.

8

Fall 2001, Lecture 31