Name of the project	: **VFS** (Virtual File System)
Technology	: Core C-Programming
User Interface	: CUI (character User Interface)
Platform	: Windows NT / Linux
Hardware Requirements	: 20 MB HDD and Processor(any one)

Basic of OS

In every operating system(OS) there are 5 major task as follows

1) Process Management
2) Memory Management
3) File Management
4) CPU scheduling
5) Hardware Abstraction

above mention points are main task of OS.

1) Process Management :

- When we write code and save it on HDD and run that program it becomes a process.
- Every run able program considered as process.
- every process has unique ID
- controls all running process that unit called as process management.
- OS stores vital information of each and every process in it's separate Data Structures(it's called as META DATA about process).

2) Memory Management :

- calculate Total memory, allocated memory, how many memory allocated for whom, this all record maintained by specific unit of OS that unit is called as memory management unit.
- memory management is the functionality of an OS which handles or manages primary memory and moves processes back and forth between main memory and disk during execution.
- Memory management keeps track of each and every memory location, regardless of either it is allocated to some process or it is free.

3) File Management :

- File is a Unformatted Uniform stream of bytes.
- In LINUX OS everything is considered as file.
- everything is storable which is called as file.
- Every OS has its own file systems.
- Every file identified by its name
- file store in particular format that is called as Magnetism.

4) CPU Scheduling :

- The process scheduling is the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a particular strategy.

5) Hardware Abstraction :

- In computer, a hardware abstraction layer(HAL) is layer of programming that allows a computer operating system to interact with a hardware device at a general of abstract level rather than at a detailed hardware level.

Description :

- The things that are always needed are saved on HHD(secondary memory).
- the things that are needed temporarily are run on RAM(Primary memory).
- Our project deals with HDD but runs in RAM due to which its Virtual.
- our project shows all information about HDD but run on RAM.

Key terms :

- **file formats** : A file format is a standard way that information is encoded for storage in a computer file. It specifies how bits are used to encode information in a digital storage medium. File formats may be either proprietary or free and may be either unpublished or open. Some file formats are designed for very particular types of data: PNG files, for example, store bitmapped images using lossless data compression.
- Every file identified by its name
- file store in particular format that is called as Magnetism.
- File store in HDD.
- HDD managed in terms of BLOCK
- In OS minimum measurable unit is called as Block.
- Block size :

  1KB = 1 Block = 1024 byte
  i) Windows - 4KB
  ii) LINUX - 1KB
- In our project minimum file size we allocate 1kb(1024 B).

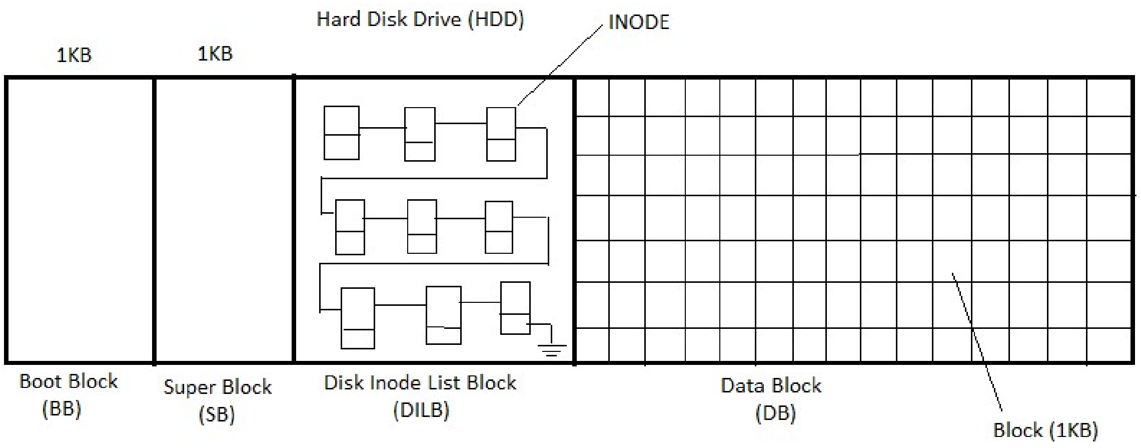
File :

- In every OS File contains own Data Structure.
- All information about file are stored in that data structure.
- It's called as Meta Data of that file.
- File structure written with name NODE and every node contains unique number that's why it called as INDEX NODE.

- The name and file in which the file name and the inode are placed can be called a directory.

➢ HDD

- ❖ HDD is divided into 4 parts.
  1) Boot Block (BB)
  2) Super Block (SB)
  3) Disk I-node List Block (DILB)
  4) Data Block (DB)



1) Boot Block :

Boot Block it is first block on HDD, it contains code which is load Operating System from HDD into RAM. In every HDD boot block size is always 1KB (1024 byte).

2) Super Block :

Super Block is second block on HDD, it's size also 1KB. A superblock is a record of the characteristics of a file system, including its size, the block size, the empty and the filled blocks and their respective counts, the size and location of the i-node tables, the disk block map and usage information, and the size of the block groups

Super block stores the information about total i-nodes and free i-nodes.

3) Disk Inode List Block :

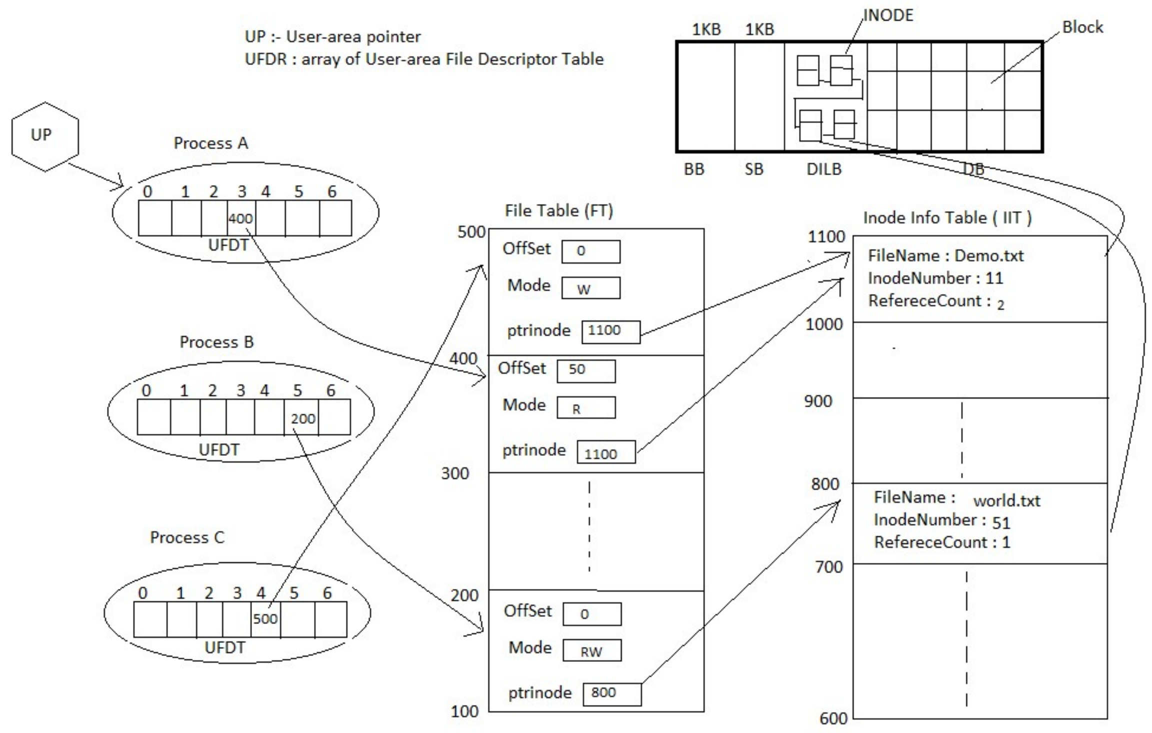
This block contains all information about file. it contains MATADATA of file. for example it stores the file name, size, actual size, file type, link count , reference count , inode number etc.

Size of DILB block is varies according to HDD size.

4) Data Block :

Data block contains actual data of file.

Size of DB block is varies according to HDD size.



➢

UP - User-Area pointer(kernel pointer).
UFDT - array of pointers(User File Descriptor Table).

How kernel access the file ?

consider above diagram:

UP is kernel pointer and it currently refers process A, in u-area of process A there is one array called as UFDTArr, in UFDTArr at index no 3 get address and go File Table. in that file table we get offset, mode & one pointer which is hold the address of file.

we goes to IIT table through pointer, in file table we get file name & INODE number in IIT table.

now we have INODE number that means we have get all information about file like file name, size, actual size, permission, type, and address of where is the data located in HDD.

Data structures used in project

Following Data structures used in project :

1) Inode (indexed node) :

Inode contains following characteristics :

- FileName : character array for hold name of file.
- InodeNumber : Unique number of inode.
- FileSize : size of file.
- FileActualSize : Actual size of file.
- FileType : Type of file(Regular or Speacific).
- Buffer : Buffer is pointer of type character which used to accept input from standard input device.
- LinkCount :
- ReferenceCount : It's shows How many process use single file.
- Permission : In which mode file is created or opened(READ or WRITE or both).
- next : next is pointer of type struct Inode.

2) SuperBlock :

Super block contains following characteristics :

- Total Inodes : contains total number of inode in file system.
- Free Inode : contains total number of free inodes.

3) FileTable :

File table contains following characteristics :

- readoffset : File Read off set, it is used for from where to start read file.
- writeoffset : File Write off set.
- count :
- mode : In which mode file is opened.
- ptrinode : it is pointer of type struct inode *.

4) UFDT (user file descriptor table)

- ptrfiletable : it is pointer of type struct FileTable *.
- it is declare as global variable.

➢ global variable

1) UFDTArr[MAXINODE];

- UFDTArr is array of type UFDT and its size is MAXINODE
- MAXINODE macro and its value is 50.
- it is declare as global variable.

2) SUPERBLOCKobj :

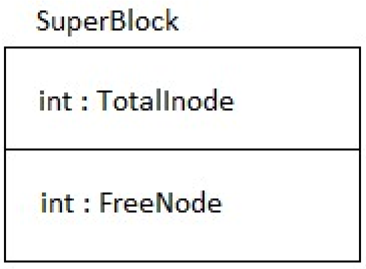
- SUPERBLOCKobj is object of SUPERBLOCK.

3) head

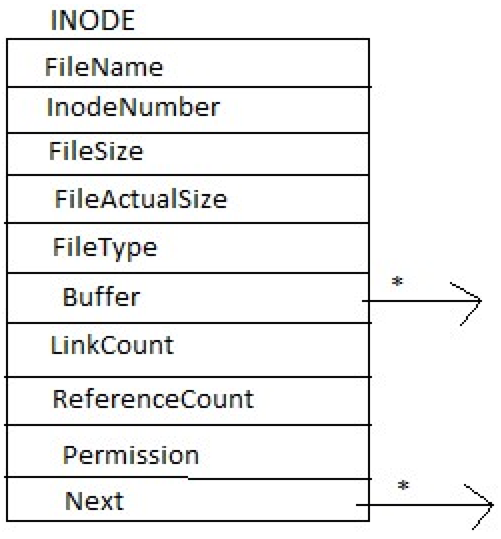
- head is pointer of type inode *.

Diagram of data structures used in the project

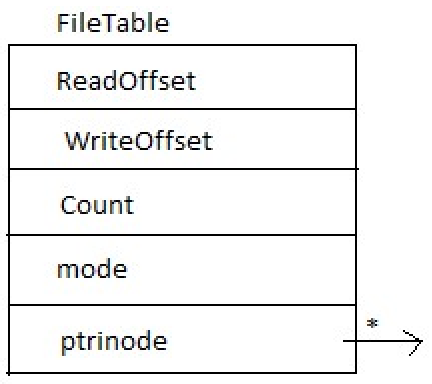
1. SUBERBLOCK



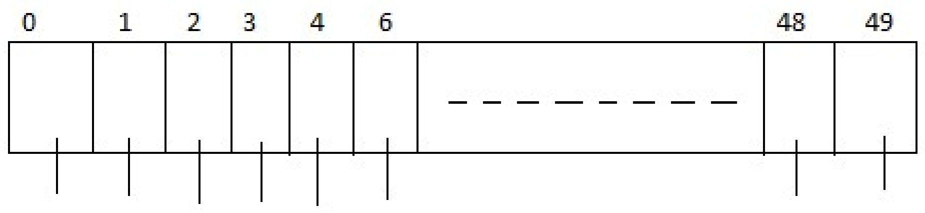
2. INODE



3. FILETABLE



4. UFDT



Functions :

1)

Function Name :	man
Input Parameter :	fileName
Return Value :	--
Description :	This function is use to display the all information about particular command.

2)

Function Name :	DisplayHelp
Input Parameter :	--
Return Value :	--
Description :	This function is use for help.

3)

Function Name :	GetFDFromName
Input Parameter :	fileName
Return Value :	file descriptor
Description :	This function accept file name and returns file descriptor integer value.

4)

Function Name :	Get_Inode
Input Parameter :	fileName
Return Value :	inode number
Description :	This function accept file name and returns Inode Number integer value.

5)

Function Name :	CreateDILB
Input Parameter :	--
Return Value :	--
Description :	This function use to create DILB block.

6)

Function Name :	initialiseSuperBlock
Input Parameter :	--
Return Value :	--
Description :	This function use to initialized DILB block.

7)

Function Name :	CreateFile
Input Parameter :	FileNme, Permission
Return Value :	iNodeNumber
Description :	This function use to create file with specific mode and return it's file number.

8)

Function Name :	rm_File
Input Parameter :	FileName
Return Value :	--
Description :	This function use to remove the file.

8)

Function Name :	WriteFile
Input Parameter :	File Descriptor, character array, size
Return Value :	size
Description :	This function use to write data into file.

9)

Function Name :	OpenFile
Input Parameter :	FileName, mode
Return Value :	file descriptor
Description :	This function use to open file read, write or read+write mode.

10)

Function Name :	CloseFileByName
Input Parameter :	File Descriptor
Return Value :	--
Description :	This function use to close the open file.

11)

Function Name :	CloseFileByName
Input Parameter :	FileName
Return Value :	0
Description :	This function use to close the open file.

12)

Function Name :	CloseAllFile
Input Parameter :	--
Return Value :	--
Description :	This function use to close the all open file.

13)

Function Name :	LseekFile
Input Parameter :	File Descriptor, size, from
Return Value :	--
Description :	used to change the location of the read/write pointer of file descriptor.

14)

Function Name :	FileStat
Input Parameter :	FileName
Return Value :	--
Description :	This function use to display all information about file.

12)

Function Name :	Truncate_File
Input Parameter :	FileName
Return Value :	--
Description :	This function use to remove all data of file.