22XX403 – OPERATING SYSTEMS

UNIT – 4 & LP – 1

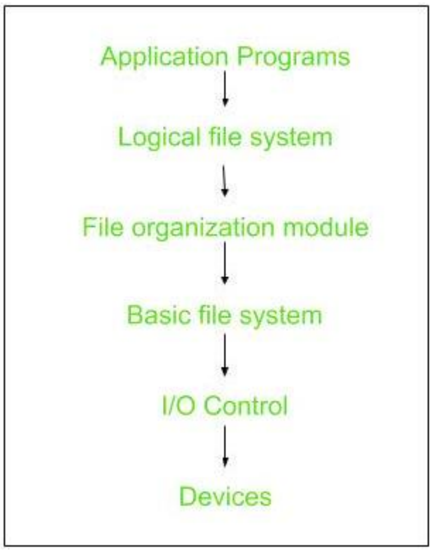
FILE SYSTEM STRUCTURES - STORAGE TECHNOLOGIES - SSD AND FLASH STORAGE OPTIMIZATION

1.1 File System Structure

File System provide efficient access to the disk by allowing data to be stored, located and retrieved in a convenient way. A file System must be able to store the file, locate the file and retrieve the file.

Most of the Operating Systems use layering approach for every task including file systems. Every layer of the file system is responsible for some activities.

The file system is divided in different layers and also the Functionalities of each layer,



- When an application program asks for a file, the first request is directed to the logical file system. The logical file system contains the Meta data of the file and directory structure. If the application program doesn't have the required permissions of the file then this layer will throw an error. Logical file systems also verify the path to the file.
- Generally, files are divided into various logical blocks. Files are to be stored in the hard disk and to be retrieved from the hard disk. Hard disk is divided into various tracks and sectors. Therefore, in order to store and retrieve the files, the logical blocks need to be mapped to physical blocks. This mapping is done by File organization module. It is also responsible for free space management.

- Once File organization module decided which physical block the application program needs, it passes this information to basic file system. The basic file system is responsible for issuing the commands to I/O control in order to fetch those blocks.
- I/O controls contain the codes by using which it can access hard disk. These codes are known as device drivers. I/O controls are also responsible for handling interrupts.

1.2 File System Operations

Common operations on file are:

Create operation, open operation, write operation, read operation, reposition operation, delete operation, truncate operation, close operation, append operation, rename operation.

Create operation on the file

To create a file in the file system, create operation is used. Create operation is the first step of the file; no other file operations can be performed without creating any file. In this operation, the file is created with no data in it. The associate applications programs in the Operating system call the file system to create a new file of any particular type. Then this file system allocated space to the file; hence the file was created with no data.

Open operation on the file

Once the file is created (with the help of create operation), it needs to be opened to perform other file processing operations. Open operation is one of the most common operations performed on the file. Open operation is simply the opening of the file.

Read operation on the file

Read operation is performed just to read the data on the file. OS(operating system) maintains a read pointer, which points to the position up to which the data has been read.

Write operation on the file.

As the name suggests, this operation is used to write the information into a file. Whenever the file length is increased by a specified value, the file pointer is repositioned after the last byte is written.

Reposition or seek operation on the file

For accessing the file randomly, a method is required to specify where to take the data; the seek operation performs this task. They seek system call repositions the file pointers from the current position to a specific position in the file (either forward or backward) depending on the user's requirement.

Truncate operation on the file

Truncate operation is used to delete the data stored inside the file. It does not delete other file attributes (it does not release the disk space).

Delete operation on the file

This operation is used to delete the file. Deleting a file will delete all the data stored inside the file, and the disk space occupied by the file will also be freed. A file needs to be deleted when it is no longer needed to free up the disk space. It is the last step of the file, as no other operations can be performed because now the file doesn't exist.

Rename operation on the file

As the name suggests, rename operation is used to change the existing file's name. We cannot assign the same name to more than one file within the same directory. This operation is used to change the file's name according to the user.

Append operation on the file

Append operation is used to add data to the end of the file. Append operation is the same as write operation on the file; the only difference is that data is added at the end of the append operation.

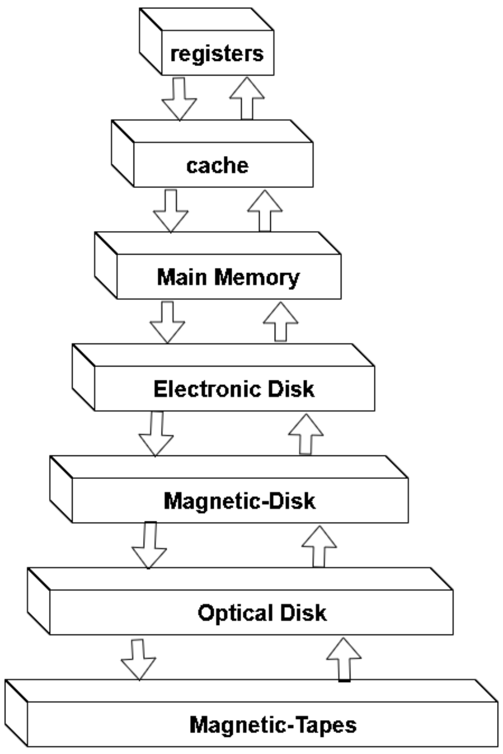
Close operation on the file

When the processing of the files finishes, it should be closed so that all the resources occupied should be released and all the changes that have been made in the file become permanent. Close operation is used to close the file. It deallocates all the internal descriptors created when the file was opened on closing.

1.3 Storage Structure in Operating Systems

Storage structure refers to the way in which data is stored and organized within a computer system. It involves the use of different storage media, such as hard disks and memory, to store and retrieve data. The storage hierarchy is a model that describes the different levels of storage used by a computer system. It ranging from the fastest and most expensive storage to the slowest and least expensive.

Storage Hierarchy



Storage Device Hierarchy

The storage hierarchy is organized in such a way that each level provides a different balance between speed, cost and capacity. The fastest and most expensive storage is located at the top of the hierarchy, while the slowest and least expensive storage is located at the bottom.

The storage hierarchy typically consists of the following levels:

Registers: Registers are the fastest type of storage in a computer system, but they are also the smallest. Registers are used to hold data that the CPU needs to access quickly, such as instructions and data being processed.

Cache Memory: Cache memory is a small amount of high-speed memory that is used to temporarily store frequently accessed data. Cache memory is faster than main memory, but it is also more expensive.

Main Memory (RAM): Main memory, also known as random access memory (RAM), is the primary storage used by a computer system. RAM is larger than cache memory, but it is also slower.

Virtual Memory: Virtual memory is a technique used by computer systems to extend the amount of available memory by using hard disk space. Virtual memory allows programs to use more memory than is physically available in the computer.

Secondary Storage: Secondary storage is used to store data and programs that are not currently being used by the computer system. Examples of secondary storage include hard disks, solid-state drives, and optical disks.

Tertiary Storage: Tertiary storage is used for long-term storage of data that is not frequently accessed. Examples of tertiary storage include tape drives and archival storage systems.

The storage hierarchy is organized in such a way that data is first stored in the fastest and most expensive storage, such as registers and cache memory, and then moved to slower and less expensive storage as needed. This allows the computer system to access data quickly when needed, while also providing a large amount of storage space for less frequently used data.

1.4 Flash Memory / Flash storage

Flash storage is a data storage technology based on high-speed, electrically programmable memory. The speed of flash storage is how got its name: It writes data and performs random I/O operations in a flash.

Flash storage uses a type of nonvolatile memory called flash memory. Flash memory is widely used to store data and code used in embedded systems. It is a non-volatile storage medium, meaning that it can retain data without a power supply. Flash memory can be electrically erased and reprogrammed and it erases data in units called blocks and rewrites data at the byte level. Flash memory is often used in systems that frequently rewrite data, such as USB flash devices or SD cards.

Flash storage uses memory cells to store data. Cells with previously written data must be erased before new data can be written. Flash storage can also come in several forms, from simple USB sticks to enterprise all-flash arrays.

Flash memory is a variation of EEPROM, or electrically erasable programmable read-only memory. EEPROM and Flash memory have many differences, with one being their reading, writing, and erasure procedures of stored data. For instance, EEPROM can read, write, and erase data at the byte level while Flash memory can also read and write at the byte level, but can only erase data at the block level.

Because erasing is a relatively slow operation and must be done before writing, performing the erase in a large block makes large write operations faster.

Types of Flash Memory

There are two types of Flash memory most commonly acknowledged: NAND and NOR Flash. NOR Flash was the first of the two to be introduced in 1988 by Intel, while NAND Flash was later introduced by Toshiba in 1989. Their main differences can be identified in their architecture.

NOR and NAND are named for the way the floating gates of the memory cells that hold data are interconnected in configurations that resemble a NOR or a NAND logic gate.

NOR Flash

NOR Flash is optimized for random access capabilities where it is able to access data in any order and doesn't require following a sequence of storage locations. In terms of its architecture, each of NOR Flash's memory cells are connected in parallel where one end of the memory cell is connected to the source line and the other end is connected to the bit line. This allows the system to access individual memory cells.

NAND Flash

Conversely, NAND Flash is optimized for high-density data storage and gives up the ability for random access capabilities. NAND Flash cells are connected, usually eight memory transistors at time, in a series to the bit line called a string. Here, the source of one cell is connected to the drain of the next one. This series connection reduces the number of ground wires and bit lines.

NAND-based Flash memory is ideal for high capacity data storage, while NOR-based Flash memory is best suited for code storage and execution, generally in small capacities.

Flash Memory Examples

Common examples of Flash memory include:

Multi-Media Card (MMC) - a Flash-based memory card standard used for solid-state storage in smartphones, digital cameras, music players, video camcorders, and personal computers. These cards store digital information such as text, pictures, audio, and video.

Solid-State Drive (SSD) - a Flash-based memory used in devices such as computers or video game consoles as a storage device that replaces hard disk drives (HDD) due to its speed and reliability.

BIOS chip – basic input/output system - is a small memory chip located on a computer's motherboard that stores instructions so the computer can perform basic functions such as booting and keyboard control.

USB Flash drive - a data storage device that includes Flash memory with an integrated USB interface. This device plugs into a PC, camera, or phone to save or transfer digital data.

Flash Memory Applications

Flash memory is widely used for storage and data transfer in consumer devices, industrial applications, and enterprise systems.

1.5 Flash Storage Optimization

Optimizing flash storage involves maximizing performance, extending the lifespan of the storage device, and ensuring efficient use of available space. Flash storage optimization is crucial for various devices, including solid-state drives (SSDs), USB drives, and memory cards.

Enable TRIM:

TRIM is a command that helps the operating system inform the SSD which data blocks are no longer in use and can be safely erased. Ensure that TRIM is enabled on your system, as it contributes to sustained SSD performance over time.

Firmware Updates:

Keep your SSD firmware up to date. Manufacturers release firmware updates to address performance issues, enhance compatibility, and improve overall reliability.

Wear Leveling:

Modern SSDs incorporate wear leveling algorithms to evenly distribute write and erase cycles across memory cells. Ensure that your SSD's wear leveling mechanisms are active to prevent premature wear on specific cells.

Over-Provisioning:

Allocate a portion of your SSD's capacity to over-provisioning. This reserved space helps maintain consistent performance and extends the lifespan of the SSD by providing additional blocks for wear leveling and bad block management.

Optimal File System:

Choose a file system optimized for flash storage. File systems like NTFS, exFAT, and APFS are designed to work well with SSDs. Research the most suitable file system for your operating system and use case.

Avoid Full Capacity:

Keep some free space on your SSD. Operating an SSD near full capacity can lead to performance degradation. Aim to leave at least 10-20% of the SSD's capacity unused to allow for efficient garbage collection and wear leveling.

Avoid Unnecessary Writes:

Minimize unnecessary write operations. This includes limiting the use of swap files, disabling hibernation (if not needed), and relocating temporary files to a separate storage device.

Temperature Management:

Keep your SSD within the recommended temperature range. Excessive heat can negatively impact SSD performance and lifespan. Ensure proper ventilation and cooling for your system.

Regular Backups:

Regularly back up your data to prevent loss in case of unexpected issues. While optimization focuses on performance and lifespan, backups are crucial for data security.

Check for Disk Errors:

Periodically check for disk errors and bad sectors. Most operating systems have built-in tools for this purpose. Timely detection and correction of errors can help maintain the health of your flash storage.

Selective Indexing:

If applicable, consider selectively indexing files and folders. Full indexing can lead to frequent writes on the storage device. Adjust indexing settings to focus on essential directories.