**Georgia Southern University**

**Allen E. Paulson College of Engineering and Computing**

**Information Technology Department**

**Dr. Atef Shalan**

**Course: IT2530-Operating Systems**

**Module 5-** I/O Devices and Storage Management – reading material

Contents and Reading Material

[**5** Module 5- I/O Devices and Storage Management 1](#_Toc89692129)

[5.1 I/O device overview 1](#_Toc89692130)

[5.2 Organization of the I/O functions 2](#_Toc89692131)

[5.3 I/O Buffering techniques 3](#_Toc89692132)

[5.4 Storage Devices 5](#_Toc89692133)

[5.5 Network Storage Systems 7](#_Toc89692134)

[5.6 Disk scheduling 8](#_Toc89692135)

[5.7 Raid Systems 9](#_Toc89692136)

# Module 5- I/O Devices and Storage Management

## I/O device overview

[I/O devices](https://www.computerhope.com/jargon/i/iodevice.htm) (AKA I/O peripherals) provide means to exchange information between a computer and external environments.

Input devices accept incoming data and convert it into binary data that is understandable to a digital computer. Examples of input devices include the keyboard, mouse, scanner, and touchscreen. Output devices convert the computer digitized signals into an environment-understandable data form. Example: Monitor, printer, etc. Input-Output peripherals allow both input and output in the same. Example: touch screens, network cards, data accusation cards. Based on their interaction with their surrounding environment, [I/O device categories](https://en.wikiversity.org/wiki/Computer_architecture/Input_Output_Organization) are as follows:

* **Human readable devices** 
  + These devices communicate with the computer user.
  + Examples include the display monitor, printer, keyboard, and mouse.
* **Machine-readable devices**
  + These devices communicate with special-purpose electronic equipment
  + Examples include hard disks, DVDs, USB sticks, sensor devices
* **Communication devices**
  + These devices communicate with other intelligent devices
  + Examples include network cards and modems

Operating Systems use device drivers to handle I/O peripherals. A device driver is a software program that can be plugged into an operating system to help interact and use the service of the I/O device.

I/O devices vary concerning their application and data representation. They also vary with respect to data transfer rate and unit of data transfer. For example, keyboard devices are used to input user writing represented as text characters into computers. Some I/O devices have low data transfer rates such as Keyboard, and mouse and some others have very high transfer rates such as ethernet cards and graphics cards. Some I/O devices transfer data in the form of a stream of bytes (e.g., keyboard and mouse), and some others transfer data in the form of a large data block, e.g., hard disks, display cards.

## Organization of the I/O functions

CPU and I/O devices need to communicate to be able to execute I/O commands, control I/O operations, and transfer I/O data. I/O devices differ concerning their methods of event trigger and data transfer. Event trigger involves allowing the CPU to detect the data arrival at the I/O device. Data transfer is moving data to/from the I/O device to the main memory. A computer CPU can detect the arrival of new data on the I/O device using one of the following methods: [polling and interrupts](https://www.tutorialspoint.com/operating_system/os_io_hardware.htm).

* **Polling I/O:** In this method,the process of periodically checking the status of the device by reading a status register that is accessible and that can be updated by the I/O devices when it receives new data. Examples: keyboard and mouse PS/2 ports.
* **Interrupts I/O:** in this method, aninterrupt is signaled to the microprocessor from a device that requests CPU attention. Examples: disk driver and network interface card.

The [communication between the computer CPU and I/O](https://www.tutorialspoint.com/operating_system/os_io_hardware.htm) device can take place through one of the following methods.

* **Special Instruction I/O:** this method uses program instructions to allow data transfer to/from the I/O devices.
* **Memory-mapped I/O:** In this method, some memory buffer is shared with the I/O devices and data transfer takes place by asynchronously reading or writing from/to this memory buffer. I/O devices interrupt the CPU when I/O operation is completed.
* **Direct memory access (DMA) I/O:** Direct memory access (DMA) method uses a hardware device to avoid or reduce the overhead caused by memory-mapped I/O device interrupts. In the DMA method, the computer CPU grants I/O module the authority to read/write memory data without involvement.

The functionality of I/O devices is provided at three levels: user, kernel, and hardware. Figure 1- describes the structure of [I/O device software](https://www.tutorialspoint.com/operating_system/os_io_software.htm) and the layers of I/O device function provisioning.

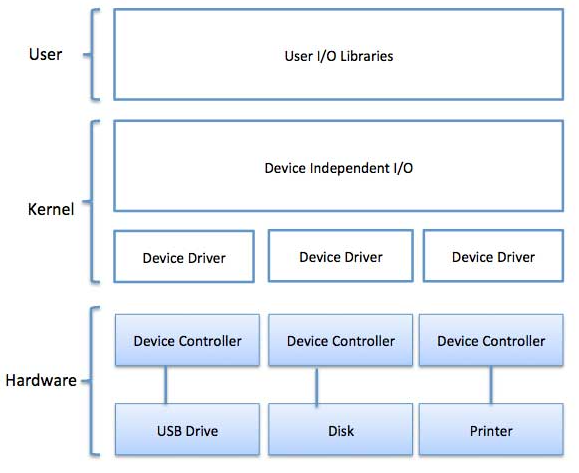


Figure - The three Layers of I/O device functions. [Visit the source of this figure](https://www.tutorialspoint.com/operating_system/os_io_software.htm).

* **User Level**: an interface to allow users and users programs to perform input and output. For example, a printing interface (or print window) through document writing software.
* **Kernel Level**: a device driver at the kernel level that can interact with the device controller at the hardware level.
* **Hardware**: the actual hardware and hardware controller that allow the hardware to interact with the environment.

I/O device driver encapsulates device-dependent code in a standard interface to the operating system kernel to help interact with the device controller and handle its functionality. Examples of the [I/O device services](https://www.tutorialspoint.com/operating_system/os_io_software.htm) include the following.

* **Scheduling:** managing I/O device requests.
* **Buffering**: maintaining shared memory areas with the I/O devices.
* **Caching:** maintaining a region of fast memory (cache) that holds copies of device data.
* **Spooling:**  maintaining an output buffer for a device.
* **Error Handling**: handlers for device errors.

## I/O Buffering techniques

I/O device buffering allows asynchronous data transfer between the I/O device and computer memory. Without a buffer, the operating must directly access the device when it needs and consequently the running process must be suspended during input and output operations. To support [I/O buffering](https://www.geeksforgeeks.org/i-o-buffering-and-its-various-techniques/), the operating system provides the following techniques.

* **Single Buffer**

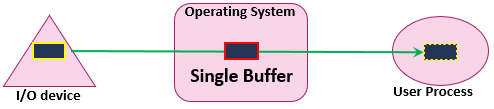
****

Figure - Single buffer I/O operations

* + A single buffer is enough storage for a single block, line, or character based on the device type (Figure 2).
  + Data transfer is performed as block-at-a-time , line-at-a-time, byte-at-a-time style.
  + I/O receives data and puts the first data block (or line or byte) in the system buffer. Then the process associated with the I/O operation transfers the buffer to its userspace and then requests another block.
  + Two blocks can be transferred simultaneously, when a data block is processed by the process, the next block is being read in.
  + To copy the data from the single buffer to user space, the operating system swaps the processes.
  + **For input operations,** the user process is suspended or blocked during input, waiting for the CPU to transfer the entire buffer to userspace.
  + **For output operations**, the user process can place a data block in the buffer and continue processing. It is only suspended if it has another block to send after the buffer is emptied.
* **Double buffer**

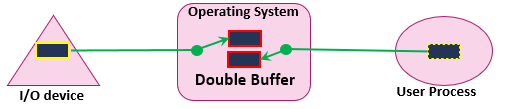
****

Figure - Double buffer I/O operations

* + Two system buffers are assigned to the I/O operations (Figure 3).
  + The operating system can swap between the two buffers: i.e., a process can transfer data to/from one buffer while the other buffer is being emptied or filled by the CPU.
  + In this case, there is no suspension during input or output operations.
  + A process may be suspended only if the buffer is full while it still needs to send more data blocks.
* **Circular buffer**

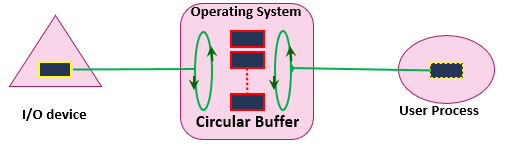
****

Figure -Circular buffer I/O operations

* + A collection of buffers is provided to the I/O operations (Figure 4).
  + Access to the set of buffer follows a circular structure that uses the producer and consumer mechanism
  + The chance of process suspension is very minor.

## Storage Devices

[A storage device](https://www.techopedia.com/definition/1119/storage-device) is a hardware system that allows data storing and extraction. Storage devices provide two main interfaces for data storage operation: mass storage and file system. Mass-storage deals with large amounts of binary data in a machine-readable format. File System Interface utilizes the concept of a file as an identifiable and specifiable resource of data. Storage devices can be classified based on the storage structure into the following.

* **Magnetic disks**

Uses magnetically coated platters to store data

* **Solid-state disks**

Uses memory-like circuit assemblies to store data persistently

* **Optical disks**

Uses laser or stamping machine to store data and uses illumination to access the data on optical data paths.

* **Tape drives**

Uses magnetic tape to store the data

* **Network storage systems**

Uses a data storage server to provide storage service over the network

* **Cloud-based storage**

Provide storage on the cloud

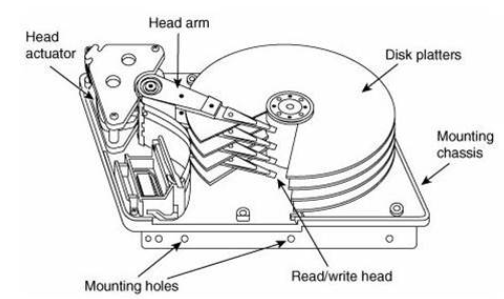


Figure -Magnetic disk structure. [Visit the source of this figure](https://dzone.com/articles/the-guts-n-glory-of-database-internals-managing-re)

Magnetic disks are the most widely used primary hard disk storage in most existing computer systems. These devices use a **Moving-head-disks** mechanism to read/write data on a circular platter that rotates at 3600 to 15000 rounds per minute (RPM). The [performance characteristics of magnetic disks](https://en.wikipedia.org/wiki/Hard_disk_drive_performance_characteristics) include access time, data transfer rate, power consumption, system reliability, and others. Access time (AKA positioning time) is the time it takes to move the head to the correct position before the drive can transfer data. The positioning time involves moving operations: moving the disk arm to the desired cylinder (**seek time**) and rotating the disk platter to bring the desired sector under the disk head (**rotational latency**).

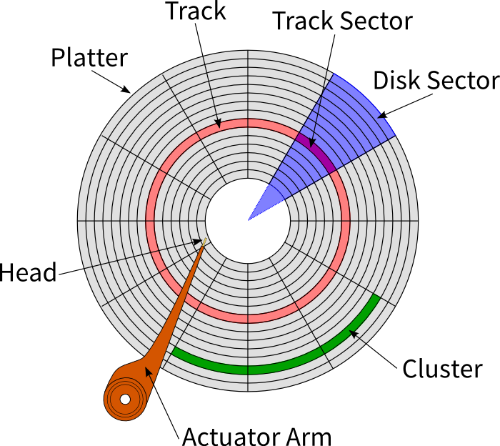


Figure - magnetic disk format. [Visit the source of this figure](https://www.cleanpng.com/png-hard-drives-disk-storage-disk-sector-solid-state-d-3089261/download-png.html).

In magnetic disks, data address space is a large one-dimensional array of logical blocks (called frames). A frame is the unit of data transfer in hard disk I/O operations. A magnetic disk is not ready for use before it is prepared using the low-level format method which creates identifiable logical blocks on the physical media. The low-level formatting maps logical blocks into the sectors of the disk sequentially. Sector 0 is located at the first sector of the first track on the outermost cylinder of the disk platter. The disk cylinders act as a continuous track medium from the outermost to the innermost cylinder.

Computer systems startup their functions using a booting process: e.g., BIOS-MBR boot method and UEFI-GPT method. The latter is a new method used in the latest versions of Windows, Linux, and Mac OS. To understand the boot sequence, we first need to define the following components.

* **Basic Input/Output System (BIOS):** the lowest software interface by which every higher-level code (i.e., bootloader and OS kernel) can communicate with the hardware.
* **Master Boot Record (MBR):** a data sector at the very bigging of the primary portioned hard disk device. MBR contains the partition table and the bootstrap code.
* **Partition table**: An index of up to four possible main partitions existing on the disk.
* **Bootstrap code**: software code that is executed by the system bios and that finds the active partition in the disk partition table and starts the execution of the bootloader procedure.
* **Bootloader procedure:** a multi-stage software process completes the booting from the active partition boot sector.
* **Active Partition boot sector:** the first sector (512 bytes) of the active partition.

The traditional [boot method (BIOS-MBR)](https://neosmart.net/wiki/mbr-boot-process/) involves six steps as follows:

1. **BIOS step:** whenthesystem is turned on, the bios initializes the boot process
2. **MBR:** The bios calls a software code (bootstrap) stored in the MBR of disk 0.
3. **Active partition:** The bootstrap loads the bootloader from the active partition boot sector.
4. **Bootloader:** the boot sector loads system configuration from files on the active partition.
5. **Boot menu**: The bootloader displays a list of OSs from a configuration file for selection.
6. **Operating system kernel**: The bootloader locates and loads the selected OS kernel.

## Network Storage Systems

Instead of connecting directly to a computer, network storage provides versatile methods for data storage over devices connected to and accessed through networks. Network storage systems include [Storage Array](https://www.dnsstuff.com/storage-array), [Network Attached Storage (NAS)](https://www.seagate.com/tech-insights/what-is-nas-master-ti/), and [Storage Area Networks (SAN)](https://hypertecdirect.com/knowledge-base/nas-vs-san/).

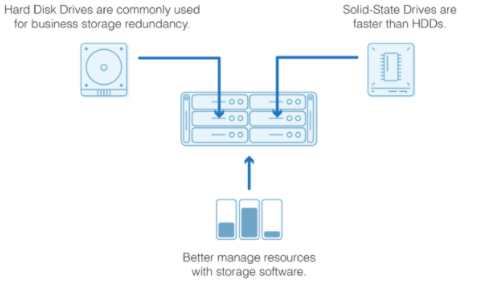


Figure 7- Storage array. [Visit the source of this figure](https://www.dnsstuff.com/storage-array).

* **Storage Array** (*aka* disk array) is a data storage system that contains multiple drives managed by a central management system (Figure 7). Storage arrays are a critical part of storage networks that is capable of storing huge amounts of data. By separating the connection and network transmission from the data storage functions, storage on these arrays can be provided as a local function to the connected computing devices. **Storage Array** **controller** is a devicethat manages the physical disk drives and provides storage features accessed through I/O ports talking to I/O busses to attached host(s).

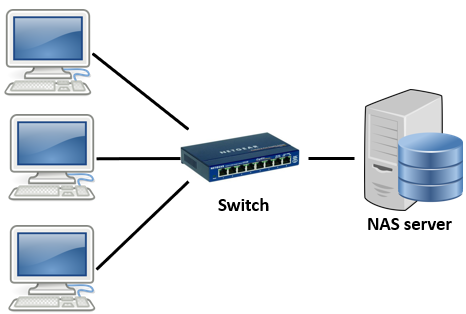


Figure 8- Network Attached Storage (NAS).

* **Network-Attached Storage (NAS)** is a storage device made available through a network connection (Figure 8). NAS devices allow data storage in and retrieval from centralized network locations that can be made available for authorized network users. Thus, they provide a flexible and scalable solution for additional storage. It also allows storage resource sharing across computer systems using the same network. Authorized computers can use a variety of [network protocols to access a NAS server](https://www.sciencedirect.com/topics/computer-science/network-attached-storage), e.g., such as Network File System (NFS) and Common Internet File System (CiFS) protocols.

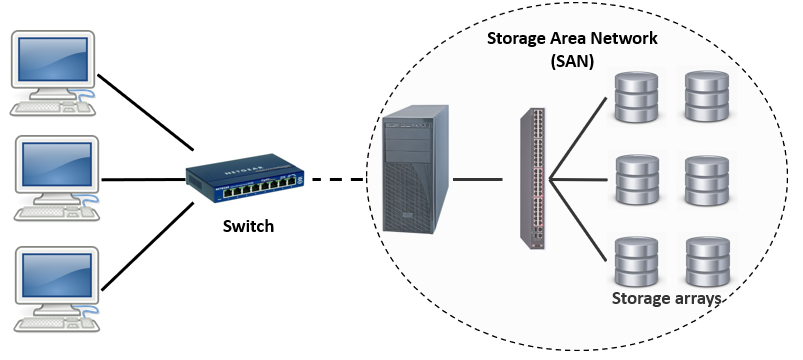


Figure -Storage Area Network

* **A storage Area Network (SAN)** is a computer network that allows organization users to access consolidated data stored on multiple storage arrays (Figure 9). SAN provides access to storage devices in specific disk arraysvia logical unit numbers (called LUN mask) for disk zoning and localization. SAN is a scalable network that allows easy plugging of additional storage devices and connecting new computer hosts.

## Disk scheduling



Figure - Disk scheduling

Hard disk is one of the busiest devices in computers. Access requests from processes incessantly pile up to form a queue of I/O requests to the secondary storage device. Disk Scheduling provides a systematic and efficient way to serve the queued disk I/O requests and meet specific performance criteria. [Disk scheduling algorithms](https://www.geeksforgeeks.org/disk-scheduling-algorithms/) vary with each based on their approach to minimize the time consumed in the mechanical operations in hard disks: seek time and rotational latency. Figure 9- shows the relationship between disk access time and these mechanical operations.

|  |
| --- |
| Disk Access Time = Seek Time +  Rotational Latency +  Transfer Time |

Figure - Disk access time.

There are a number of scheduling algorithms as follows.

* **First Comes First Serve (FCFS)** addresses requests in the order they arrive in the disk queue.
* **Shortest Seek Time First (SSTF)** selects the request with the minimum seek time from the I/O queue based on the current head position.
* **SCAN algorithm** takes into consideration the direction of disk head motion. Therefore, it keeps selecting the nearest requests towards one direction and then turns back to other directions taking the remaining requests based on the shortest seek time.
* **C-SCAN algorithm** is like SCAN except that it serves the I/O requests as it moves only in one direction. I.e., it selected the shortest seek time first towards one direction and when it reaches the end it starts again from the beginning.
* **Lock and C-Lock algorithms** are like SCAN and C-SCAN except for, they don’t have to come exactly to the end (The first of last cylinder in the disk), instead, they keep moving till the position of the last I/O request in that direction and then they turn back (as in Lock) or start from the bigging (C-Lock).

See the illustrative example using a specific I/O queue example on [this page](https://www.geeksforgeeks.org/disk-scheduling-algorithms/).

## Raid Systems

Reliable storage involves the use of multiple software and hardware techniques to assure data integrity on the storage devices. Some reliable storage mechanisms include the following.

* **Redundant Array of Independent Disks (RAID)** is adata storage technology that assures data integrity by using disk arrays with redundancy.
* **Backup & Recovery** mechanisms use copies of important data to reinstate its integrity in case of any damage or loss of data.
* **Snapshot** means a system view backup at a specific point of time before taking some action or making a critical change to the system.
* **Replication** is a systematic duplication of data change activities separate data storage sites

**Redundant Array of Independent Disks (RAID)** is adata storage technology that assures data integrity by combining multiple physical disk drives into one or more logical drives and creating redundant copies of data on various structures of disk arrays. We call the disk array structures used in these system [RAID levels](https://searchstorage.techtarget.com/definition/RAID). The different RAID levels vary concerning third target quality criteria: data integrity, access speed, and storage capacity. All RAID system structures maintain the following design criteria.

* **Single drive access:** disk arrays are viewed as a single logical drive.
* **Data distribution:** across multiple disk drives.
* **Data recoverability:** guaranteed by using some forms of redundancy.

RAID solutions come in seven different levels as follows

* **RAID level 0** the data evenly across multiple disks (AKA **striping**) to allow using them as one storage unit.
* **RAID level 1** maintains a redundant copy of each disk (AKA **mirroring**)
* **RAID level 2** provides an efficient and low-cost model using hamming code redundancy (in only three disks) instead of a data duplication model.
* **RAID level 3** provides **high data transfer rates** by using the bit-parity redundancy model in one disk only.
* **RAID level 4** provides **high I/O request rates** by using the block-parity redundancy model.
* **RAID level 5** provides interleaved parity model and avoids a Single point of failure.
* **RAID level 6** provides a dual interleaved parity model and extremely high system availability.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_