9...

I/O System

Objectives...

- To introduce concept of I/O System.
- To learn about I/O Hardware.
- To study Application of I/O Interface and Kernel I/O Subsystem.
- To get information of Disk Scheduling algorithms such as FCFS, Shortest Seek Time First, SCAN (Elevator Disk Algorithm), C-SCAN, C-LOOK.

9.1 INTRODUCTION

- Operating system controls all the computers I/O devices, it issues command to devices, catch interrupts and handler errors. It should provide the interface between device and rest of the system.
- The control of devices connected to the computer is a major concern of operating-system designers because I/O devices vary so widely in their function and speed varied methods are needed to control them and these methods form the I/O subsystem of the kernel, which separates the rest of the kernel from the complexities of managing I/O devices.
- The role of the operating system in computer I/O is to manage and control I/O operations and I/O devices.
- I/O device technology exhibits two conflicting trends one hand, we see increasing standardization of software and hardware interfaces. This trend helps us to incorporate improved device generations into existing computers and operating systems and on the other hand, we see an increasingly broad variety of I/O devices.
- Some new devices are so unlike previous devices that it is a challenge to incorporate them into our computers and operating systems. This challenge is met by a combination of hardware and software techniques.
- The basic I/O hardware elements, such as buses, ports and device controllers, accommodate a wide variety of I/O devices. To encapsulate the details and oddities of different devices, the kernel of an operating system is structured to use device driver modules.
- The device drivers present a uniform device-access interface to the I/O subsystem, much as system calls provide a standard interface between the application and the operating system.

- The I/O devices with computer system can be roughly group into three categories:
  - **Human readable:** Use for communicating with the computer uses.
    Examples: printer, keyboard.
  - **Machine readable:** Use for communicating with electronic equipment.
    Examples: disk and tape drives, sensors, controllers.
  - **Communication:** Use for communicating with remote devices.
    Example: modems.
- These are three techniques for performing I/O:
  - **Programmed I/O:** The processor issues an I/O command, on behalf of process, to an I/O module; that process then busy waits for the operation to be completed before proceeding.
  - **Interrupt-driven I/O:** The processor issues an I/O command, on behalf of process, continues to execute next instructions, and is interrupted by the I/O module when the latter has completed its work.
  - **Direct Memory Access (DMA):** A DMA module controls the exchange of data between main memory and an I/O module. The processor sends a request for the transfer of a block of data to the DMA module and is interrupted only after the entire block has been transferred I/O system organization.
- In modern operating system, device management is implemented either through the interaction of device driver and interrupt routine called interrupt driven I/O or wholly within device driver if interrupt are not used called direct I/O with polling.
- Fig. 9.1 shows the unit involved in I/O operation for both approaches.

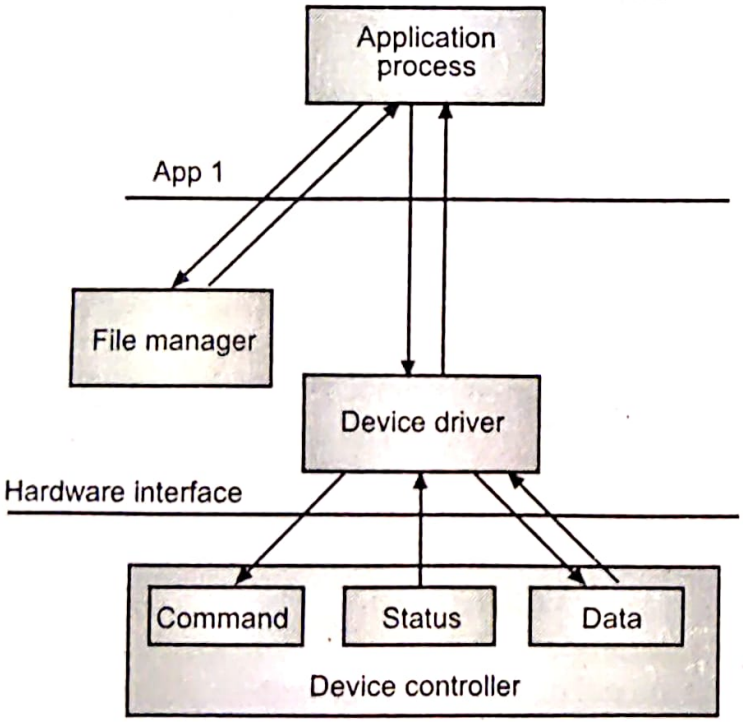


Fig. 9.1: I/O Operations

- An application process uses device by issuing commands and exchanging data with the device management device drives. The device driver has two major responsibilities.

- Implement an abstract application programming interface (API) to the application process.
- Provide device dependent operations to issue appropriate command to implement functions defined on the API.

9.2 I/O HARDWARE [S-18, W-22]

- Basically, computers operate a great many kinds of devices. Most fit into the general categories of storage devices such as disks, tapes etc. transmission devices such as network cards, modems etc. and human-interface devices such as screen, keyboard, mouse and so on.
- Devices communicate with the computer via signals sent over wires or through the air.
- Devices connect with the computer through **ports**, e.g. a serial or parallel port.
- **Bus:** If a device uses common set of wires it is called as bus. A Bus is nothing but set of wires with protocol that specifies set of messages that can be sent over the wires.
- Buses are widely used in computer architecture, and they vary in signaling methods, their speed, throughput and connection methods.
- A Common PC bus structure is given in Fig. 9.2.
- **PCI bus** connects processor-memory subsystem to the fast devices and expansion bus that connects relatively slow devices like keyboard, serial and USB ports.
- Disks are connected with **SCSI bus** plugged into SCSI Controller. Other buses are also used to connect main parts of computer like PCI-X bus and PCI-e (express) bus.
- A **Controller** is collection of electronics that operate on port, bus or device.
- A **Serial Port controller** is simple device controller. It is a single chip in computer that controls signals on wire of serial port.
- **SCSI controllers** are complex; it is often implemented as separate circuit board, that is plugged in to computer. It typically contains processor, microcode and some private memory to enable it to process the SCSI protocol message.
- **Disk controller** has one or more registers for data and control signals. The processor communicates with the controller by reading and writing bit patterns in these registers. One way in which this communication can occur is through the use of special I/O instructions that specify the transfer of a byte or word to an I/O port address.
- Another technique for communicating with devices is ***memory-mapped I/O.***
- In this case, the I/O instruction triggers bus lines to select the proper device and to move bits into or out of a device register. Alternatively, the device controller can support memory-mapped I/O. In this case, the device-control registers are mapped into the address space of the processor. The CPU executes I/O requests using the standard data-transfer instructions to read and write the device-control registers.
- Some computer systems use both techniques. For instance, PCs use I/O instructions to control some devices and memory-mapped I/O to control others.

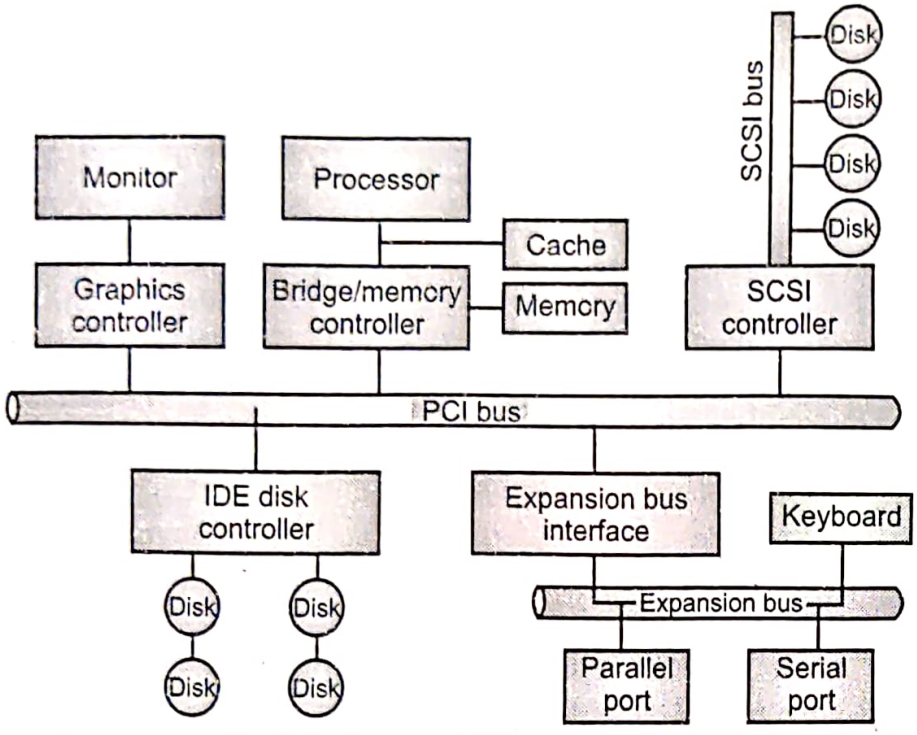


Fig. 9.2: A typical PC bus structure

- Table 9.1 shows the usual I/O port addresses for PCs. The graphics controller has I/O ports for basic control operations, but the controller has a large memory mapped region to hold screen contents.
- The process sends output to the screen by writing data into the memory-mapped region then the controller generates the screen image based on the contents of this memory. This technique is simple and easy to use.

Table 9.1: Most common Device I/O Port Address Ranges

I/O address range (hexadecimal)	Device
000-00F	DMA controller
020-021	Interrupt controller
040-043	Timer
200-20F	Game controller
2F8-2FF	Serial port (Secondary)
320-32F	Hard-disk controller
378-37F	Parallel port
3D0-3DF	Graphics controller
3F0-3F7	Diskette-drive controller
3F8-3FF	Serial port (primary)

- An I/O port typically consists of four registers called, status, control, data-in, and data-out registers.
  1. **The data-in register:** It is read by the host to get input.
  2. **The data-out register:** It is written by the host to send output.

3. **The status register:** It contains bits that can be read by the host. These bits indicate states.
4. **The control register:** It can be written by the host to start a command or to change the mode of a device.

- The data registers are typically 1 to 4 bytes in size. Some controllers have FIFO chips that can hold several bytes of input or output data to expand the capacity of the controller beyond the size of the data register.

Direct I/O with Polling:

- In direct I/O method, to do I/O, the CPU is responsible for transferring the data between the primary memory and the device controller data registers.
- While managing I/O, the device manager may poll the device busy-done flags or use interrupts to detect the operations completion.
- Steps required to do an input operation using polling (Refer Fig. 9.3):
  (i) The application process requests a read operation.
  (ii) The device driver queries the status register to determine if the device is idle. If the device is busy, the driver waits for it to become idle.
  (iii) The driver stores an input command into the controller's command register, thereby starting the device.
  (iv) The driver repeatedly reads the status register while waiting for the device to complete its operations.
  (v) The driver copies the contents of the controller's data registers into the user process space.
- The steps to perform an output operation are as follows:
  1. The application process requests a write operation.
  2. The device driver queries the status register to determine if the device is idle. If device is busy, the driver waits for it to become idle.

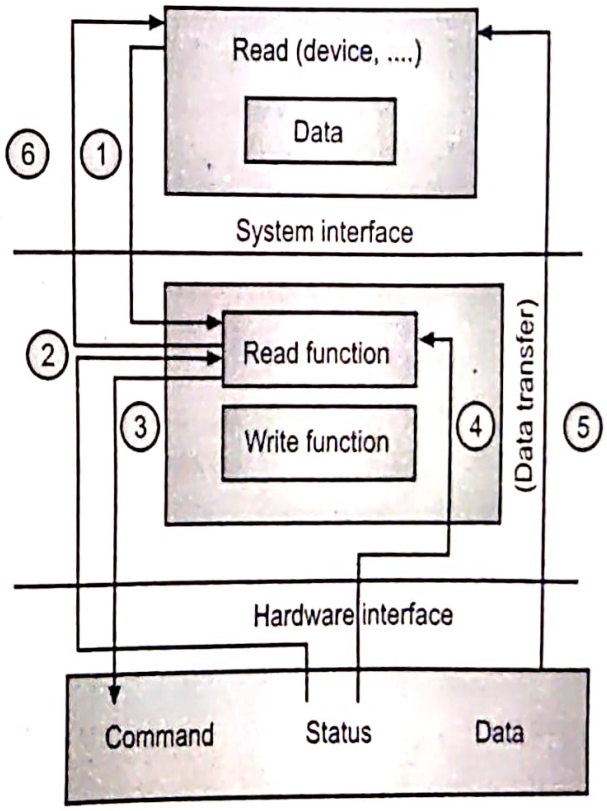


Fig. 9.3: I/O operations using Polling

3. The driver copies data from user space memory to the controller's data registers.
4. The driver stores an O/P command into the command register, thereby starting the device.
5. The driver repeatedly read the status register while waiting for the device to complete its operation.

Interrupts: [S-19, 22; W-22]

- The motivation for incorporating interrupt into the computer hardware is to eliminate the need for the device driver to constantly poll the controller status register.
- In the scenario using interrupts, the device management functionality is partitioned into four different parts:
  1. The device driver that initiates the I/O operation.
  2. Device status table.
  3. Interrupt handler.
  4. Device handler.
- The following are the steps for performing an input instruction in a system by using interrupts, (Refer Fig. 9.4).
  1. The application process requests a read operation.
  2. The top half of device driver queries the status register to determine if device is idle. If the device is busy, the driver waits for the device to become idle.
  3. The driver stores an input command into the controllers command register, thereby starting the device.
  4. When the top half of the device driver completes it work, it saves information regarding the operation that it began in the Device Status Table. This table contains an entry for each device in the system. The top half of the driver writes information into the entry for the device it is using such as the return address of the original call and any special parameters for the I/O operation. The CPU then can be used by another program, so the device manager invokes the scheduler past of the process manager. It then terminates.
  5. Eventually the device completes the operation and interrupts the CPU, thereby causing the interrupt handler to run.
  6. The interrupt handler determines which device causes the interrupt. It then branches to the device handler for that device.
  7. The device handler retrieves the pending I/O status information from the device status table.
  8. The device handler copies the contents of the controller's data registers into the user processes space.
  9. The device handler invoked by the application process return control to the application process. The output process behaves similarly.

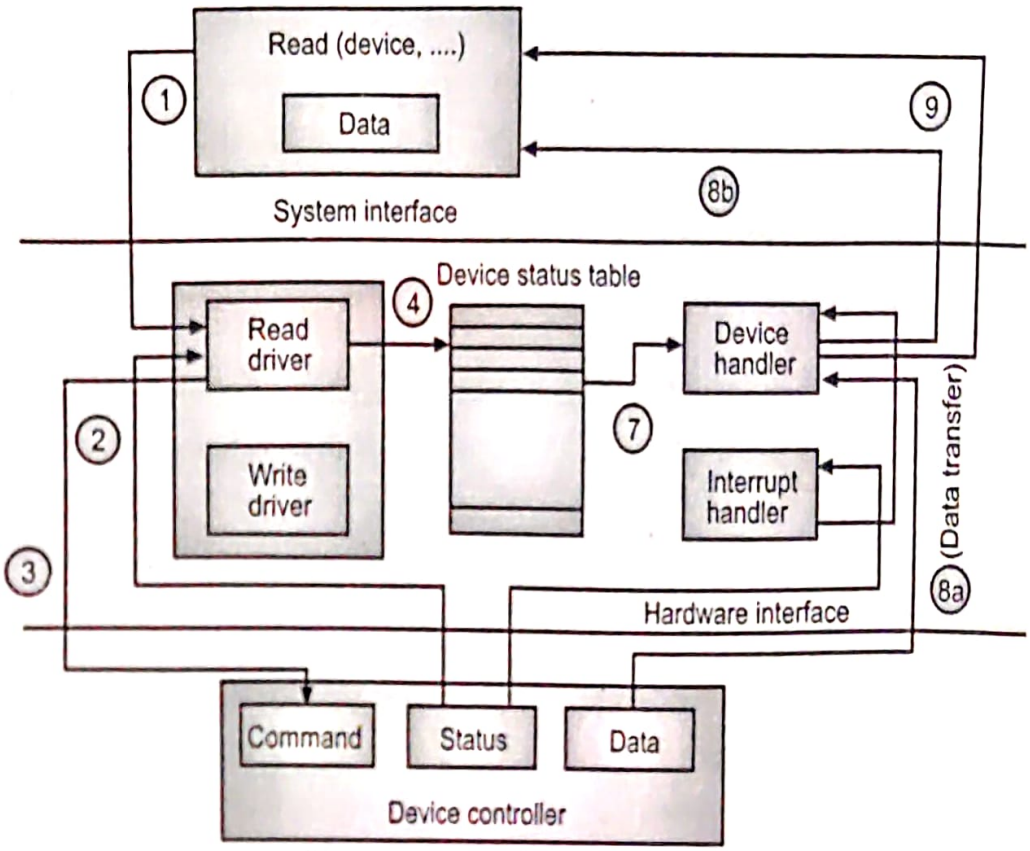


Fig. 9.4: I/O operations using interrupt

Direct Memory Access (DMA):

S-19

- For a device that transfers large quantities of data, such as disk drive, it is convenient to use an expensive general purpose processor to watch status bit and to feed data into controller register one byte at a time.
- Many computers avoid burdening the main CPU by off-loading some of this work to a special purpose processor called **Direct Memory Access (DMA) controller.**
- The DMA controller has access to system bus independent of the CPU as shown in Fig. 9.5.

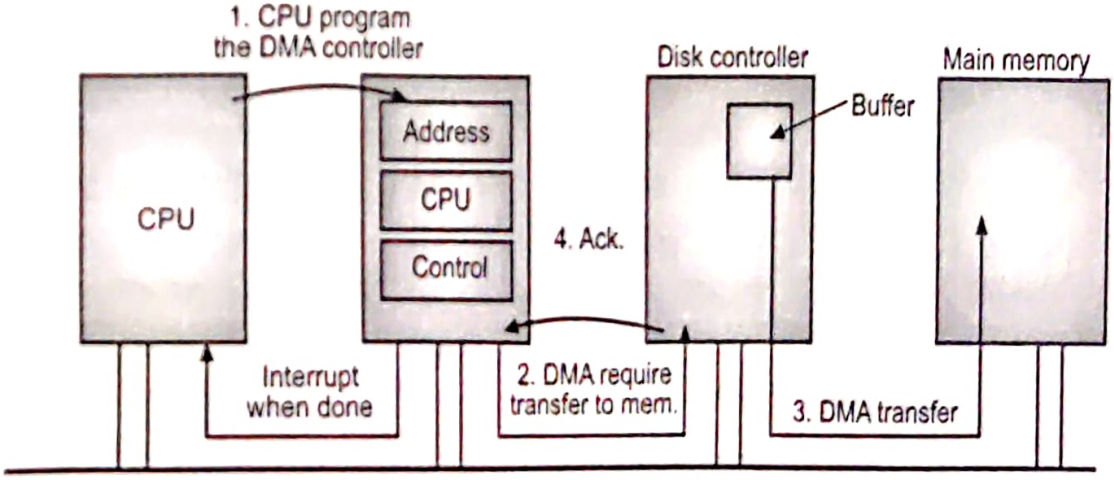


Fig. 9.5: DMA Controller

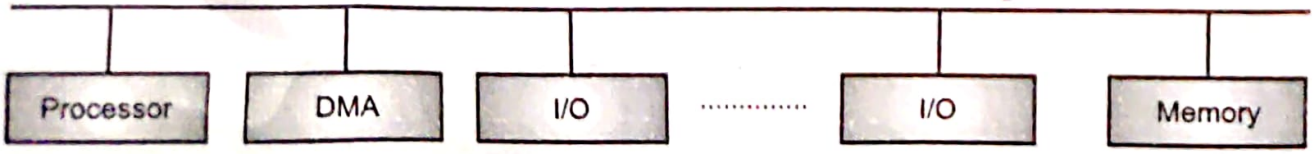
- DMA contains several registers that can be written and read by the CPU.
- This includes a memory address register, a byte count register and one or more control registers.
- The control registers specify the I/O port to use, the direction of the transfer, the transfer unit and the number of bytes to transfer in one burst.

Steps in DMA transfer:

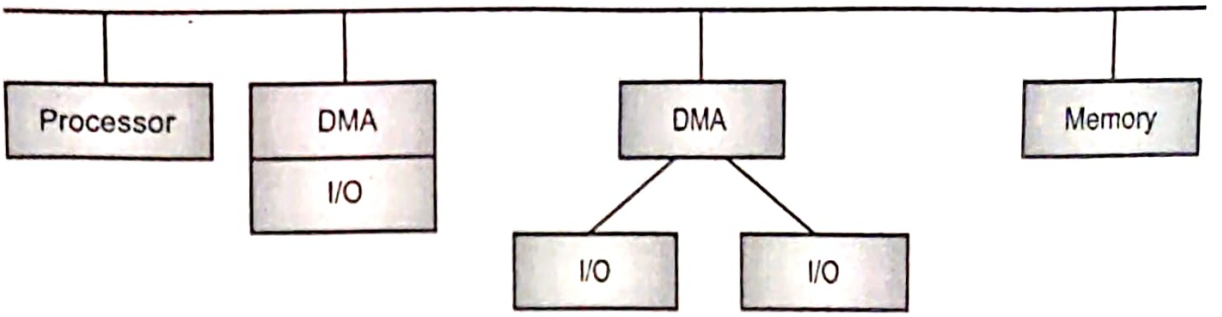
1. The CPU programs the DMA controller by setting its registers, so it knows what to transfer. It also issues a command to the disk controller to read data from the disk into it internal buffer and verify the checksum. When valid data are in the disk controller's buffer, DMA can begin.
2. The DMA controller initiates the transfer by issuing read request over the bus to the disk controller.
3. The memory address to write to is on the bus address lines so when the disk controller fetches the next word from its internal buffer. It knows where to write it. The write to memory is another standard bus cycle.
4. When the write is complete, the disk controller sends an acknowledgment signal to the disk controller, also over the bus.
5. The DMA controller then increments the memory address to use and decrements the byte count. If the byte count is still greater than 0, step 2 through 4 are repeated until the count reaches 0.
6. At that time, the DMA controller interrupts the CPU to let it know that the transfer is now complete.

DMA Mechanism:

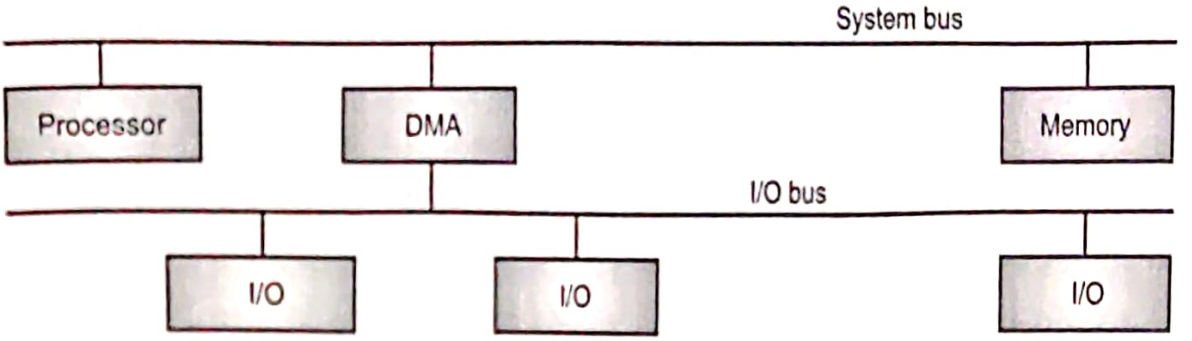
- The DMA mechanism can be configured in a variety of ways. Some possibilities are shown in Fig. 9.6.
- In the first example (Fig 9.6(a)), all modules share the same system bus. The DMA module, acting as a surrogate processor, uses programmed I/O to exchange data between memory and I/O module through DMA module.
- This configuration may be inexpensive but is inefficient. As with processor controlled programmed I/O, each transfer of a word consumes two bus cycles.
- The number of required bus cycles can be substantially by integrating the DMA and I/O functions.
- As in Fig. 9.6(b) indicates, this means that there is a path between the DMA module and one or more I/O modules that does not include the system bus.
- The DMA logic may actually be a part of an I/O module, or it may be separate module that controls one or more I/O modules.
- This concept can be taken on one step further by connecting I/O modules to the DMA module using an I/O bus, (Fig. 9.6(c)). This reduces the number of I/O interfaces in the DMA module to one and provides for an easily expandable configuration.



(a) Single bus, detached DMA



(b) Single bus, integrated DMA - I/O system bus



(c) I/O bus

Fig. 9.6: Configuration of DMA Mechanism

9.3 APPLICATION I/O INTERFACE

- Application I/O Interface represents the structuring techniques and interfaces for the operating system to enable I/O devices to be treated in a standard, uniform way.
- The actual differences lies kernel level modules called **device drivers** which are custom tailored to corresponding devices but show one of the standard interfaces to applications. The purpose of the device-driver layer is to hide the differences among device controllers from the I/O subsystem of the kernel, such as the I/O system calls.
- Fig. 9.7 shows the I/O-related portions of the kernel are structured in software layers.
- The purpose of the device-driver layer is to hide the differences among device controllers from the I/O subsystem of the kernel, much as the I/O system calls encapsulate the behavior of devices in a few generic classes that hide hardware differences from applications.
- Making the I/O subsystem independent of the hardware simplifies the job of the operating-system developer. It also benefits the hardware manufacturers.
- They either design new devices to be compatible with an existing host controller interface or they write device drivers to interface the new hardware to popular operating systems for this reason we can attach new peripherals to a computer without waiting for the operating-system vendor to develop support code.

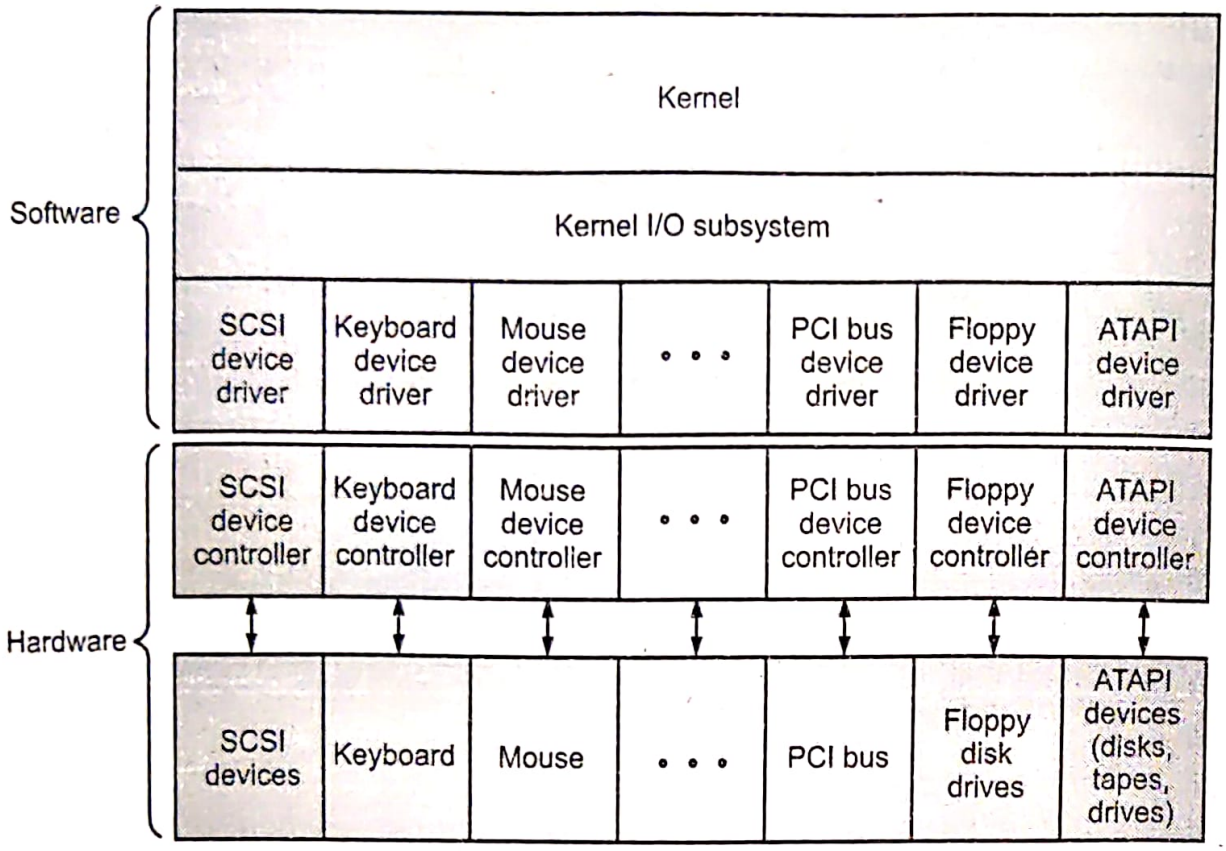


Fig. 9.7: A Kernel I/O structure

- Unfortunately, for device-hardware manufacturers, each and every type of operating system has its own standards for the device-driver interface.
- A given device may ship with multiple device drivers for instance, drivers for Windows NT/2000, MS-DOS, Windows 95/98/2000 and Solaris. Devices vary on many dimensions, as shown in Table 9.2.
  - **Character-stream or Block:** This device transfers bytes one by one, whereas a block device transfers a block of bytes as a unit.
  - **Sequential or Random access:** This device transfers data in a fixed order determined by the device, whereas the user of a random-access device can instruct the device to seek to any of the available data storage locations.
  - **Synchronous or Asynchronous:** This device performs data transfers with predictable response times. An asynchronous device exhibits irregular response times.
  - **Sharable or Dedicated:** This device can be used concurrently by several processes or threads; a dedicated device cannot.
  - **Speed of operation:** Device speeds range from a few bytes per second to a few gigabytes per second.
  - **Read-write, Read only or Write only:** Some devices perform both input and output, but others support only one data transfer direction.

Table 9.2: Characteristics of I/O devices

Aspect	Variation	Example
data-transfer mode	character block	terminal disk
access method	sequential random	modem CD-ROM
transfer schedule	synchronous asynchronous	tape keyboard
device speed	latency seek time transfer rate delay between operations	
I/O direction	read only write only read-write	CD-ROM graphics controller disk

- Most devices can be characterized as either Block I/O, Character I/O, Memory mapped file access, or Network sockets.
- Operating systems also provide special system calls to access a few additional devices, such as a time-of-day clock and a timer. Some operating systems provide a set of system calls for graphical display, video and audio devices.
- Most operating systems also have an escape or back door, which allows applications to send commands directly to device drivers if needed.

9.4 KERNEL I/O SUBSYSTEM

- Kernel provides different services related with I/O such as scheduling, caching, spooling, device reservation, and error handling.
- The I/O sub system is also responsible for protecting itself from errant processes and malicious users.

1. I/O Scheduling:

- It means to determine a good order in which to execute input/output requests called **Disk Scheduling**. Disk scheduling is also known as I/O scheduling.
- Scheduling I/O requests can greatly improve overall efficiency. Priorities can also play important role in request scheduling. It can reduce average waiting time for I/O completion process.
- By this device access, permissions will be fair to all jobs, when application issues a blocking I/O system call, the request is placed on the job queue for that device. The order of the queue rearranges by the kernel I/O scheduler to improve the overall system efficiency.

2. Buffering:

- A buffer is an area of memory that stores data when they are moved between two devices or between a device and an application.
- Buffering is done for following three reasons:
  - **Speed differences between two devices:** A slow device may write data into a buffer, and when the buffer is full, the entire buffer is sent to the fast device all at once. So that the slow device still has somewhere to write while this is going on, a second buffer is used, and the two buffers alternate as each becomes full. This is known as **double buffering.**
  - **Data transfer size differences:** To provide adaptation for data that have different data-transfer sizes.
  - **To support copy semantics:** Third use of buffering is to support copy semantics for the application I/O, "copy semantic" means, suppose that an application wants to write data on a disk that is stored in its buffer. It calls the write() system's call, providing a pointer to the buffer and the integer specifying the number of bytes to write.

3. Caching:

- A cache is a fast memory area that contains copy data. Access to a cached copy is more efficient than access to the original data. For example, instructions from the current process are stored on disk, cached in physical memory, and then copied again into the secondary and primary cache of the CPU.
- The difference between a buffer and an cache is that the buffer can store the only data information whereas a cache by definition only stores a data from a place to be accessed faster.
- Caching and buffering are two different functions, but sometimes a memory area can be used for both. For example, to save on semantics copy and make scheduling I/O efficient, the operating system uses a buffer in main memory to store data.
  - This buffer is also used as cache, to improve the efficiency of I/O for files that are shared by several applications, or that are being read and written repeatedly.
  - When the kernel receives an I/O file request, the kernel accesses the buffer cache to see whether memory area is available in main memory.
  - If it is available, a physical disk I/O can be avoided or it is not used. Disk writing also accumulates into the buffer cache for few seconds, so large transfer will be collected to streamline the writing schedule.

4. Device Spooling and Reservation:

- A spool is a buffer that stores output for a device, such as a printer, that cannot accept interleaved data streams. Even though the printer can only serve one job at a time, some applications can require the printer to print, without having to get their output mixed together.

- The operating system will solve this problem by intercepting all output to the printer. Each application output has been spooled to a different disk file. When an application finishes printing, the spooling system will continue to the next queue. In some operating systems, spooling is handled by a process daemon system. On other operating systems, this system is handled by an in-kernel thread.
- For some devices, such as screen drives and printers, it cannot multiplex I/O requests from some applications. Spooling is one way to overcome this problem. Another way is to divide coordination for multiple concurrent.
- Some operating systems provide support for exclusive device access, by allocating processes to idle devices and removing devices that are no longer needed. Other operating systems impose limits on a file to handle this device. Many operating systems provide functions that make the process of handling coordinates exclusive access among them.

5. **Error Handling:**

- An operating system that uses protected memory can guard against many possible errors due to hardware or applications. Devices and I/O transfers can fail in many ways, because of transient reasons, such as overloaded on the network, or permanent reasons such as damage to the disk controller.
- Operating systems can often compensate for transient errors. Like, a read error on the disk will cause a reread and a transmission error on the network will result in a re transmission if the protocol is known. However, for permanent errors, the operating system in general will not be able to restore the situation to normal.
- A general rule, an I/O system will return one bit of information about the status of the call, which will indicate whether the process is successful or failed.
- The operating system on UNIX uses an additional integer called errno to return an error code of about 1 out of 100 values that indicate the cause of the error. However, some hardware can provide detailed error information, even though many operating systems do not support this facility.

6. **I/O Protection:**

- Errors and the issue of protection are closely related. A user process may attempt to issue illegal I/O instructions to disrupt the normal function of a system. We can use the various mechanisms to ensure that such disruption cannot take place in the system. To prevent illegal I/O access, we define all I/O instructions to be privileged instructions. The user cannot issue I/O instruction directly.

9.5 DISK SCHEDULING [W-18]

- Over many years, the increase in the speed of processor and main memory put impact on disk access.
- This gap between speed of processor, main memory and disk is expected to continue into the foreseeable future. Thus, the performance of disk storage subsystem is important and much research has goes into schemes for improving that performance.
- Because the performance of the disk system is tied closely to file system design issues, discuss in next sections.

Disk Performance Parameters:

- The actual details of disk I/O operation depend on the computer system, the operating system and the nature of the I/O channel and disk controller hardware.
- A general timing diagram of disk I/O transfer is shown in Fig. 9.8.

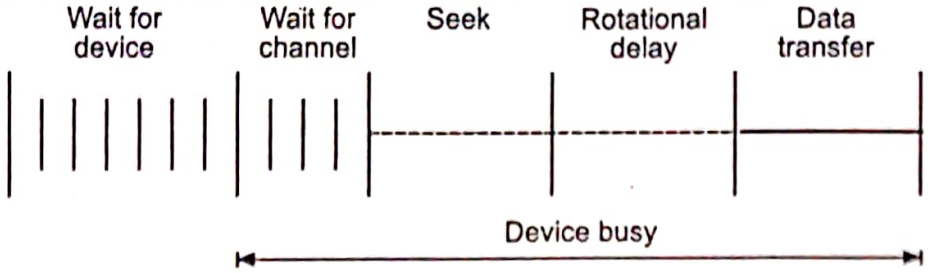


Fig. 9.8: Timing of a Disk I/O Transfer

- When the disk drive is operating, the disk is rotating at constant speed. To read or write, the head must position at desired track and at beginning of desire sector on the track.
- Track selection involves moving the head in movable head system or electronically selecting one head on a fixed-head system. On a movable-head system, the time it takes to position at track is known as seek time.
- In either case, once the track is selected the disk controller waits until the appropriate sector rotates to line up with the head. The time it takes for beginning of the sector to reach the head is known as rotational delay or rotational latency.
- The sum of the seek time, if any and the rotational delay is the access time.
- Once, head is positioned, the read or write operation is then performed as the sector moves under the head; this is the data transfer portion of the operation.
- In addition to the access time and transfer time, there are several queuing delay associate with I/O operation. When a process issues an I/O request, it must first wait in a queue for the device to be available.
- At that time a device is assigned to the process. If device shares a single channel or set of I/O channels with other disk drives, then there may be an additional wait for the channel to be available. At that point, seek is performed to begin disk access.

Seek Time: **[W-22; S-23]**

- Seek time is the time required to move the disk arm to the required tack. The seek time consists of two key component – the initial startup time and the time taken to traverse the tracks that have to be crossed once the access is up to speed.
- Unfortunately, the traversal time is not linear and settling time. Much improvement comes from smaller and lighter disk components.

Rotational Delay:

- Magnetic disk, other than floppy disk, have rotational speed in the range 5400 to 10,000 r.p.m., the latter equivalent to one revolution per 6 ms. Thus, at 10,000 r.p.m., the average rotational delay will be 3 ms.
- Floppy disk rotates at between 300 and 600 r.p.m. The average delay will be between 100 and 200 ms.

Transfer Time:

- The transfer time to or from the disk depends on the rotation speed of the disk in the following manner:

$$T = \frac{b}{rN}$$

where, T = Transfer time

b = Number of bytes to be transferred

N = Number of bytes on a track

r = Rotation speed in revolution per second

Thus, the Total Average Access Time can be expressed as,

$$T_a = T_s + \frac{1}{2r} + \frac{1}{rN}$$

where, T_s is the average seek time.

Disk Access Time:

Disk access time is given as,

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

Disk Response Time:

- It is the average of time spent by each request waiting for the I/O operation.

Disk Scheduling Algorithm:

- In Multiprogramming environment, the operating system maintains a queue of requests for each I/O device. So, for a single disk, there will be a number of I/O request from various processes in queue.
- If we selected request from the queue in random order, then tracks to be visited will occur randomly, giving the worst possible performance. Thus, there is the necessity of good algorithms.

9.5.1 FCFS (First Come First Serve) [W-18]

- The simplest scheduling algorithm is FCFS (First Come First Serve) which simply means that process request from queue in sequential order. This algorithm is quite simple.

Example 1: Fig. 9.9 illustrates the disk arm movement with FCFS. In this example, we assume a disk with 200 tracks and then the disk request queue has random requests in it. The requested tracks, in the order received are 55, 58, 39, 19, 90, 160, 150, 38, 184. Initial position of head is at 100 tracks.

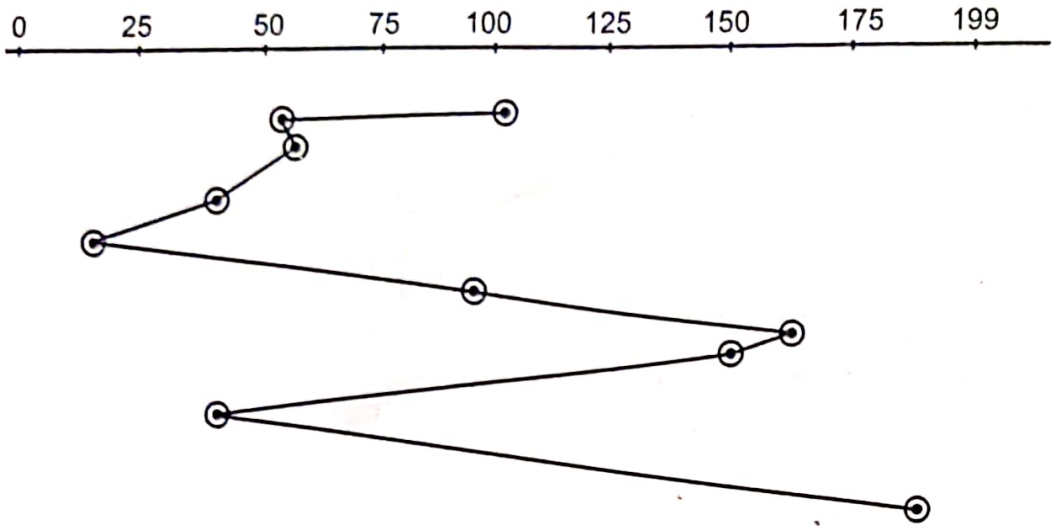


Fig. 9.9: FCFS (Starting at Track 100)

Solution: The average seek length is,

Next Track Accessed	Number of Track Traversed
55	45
58	3
39	19
18	21
90	72
160	70
150	10
38	112
184	146
Average Seek Length	**55.3**

With FIFO, if there are few processes the require access and if many of the requests are to clustered file sectors, then we can hope for good performance.

However, this technique will often approach random scheduling in performance, if these are many processes competing for the disk. Thus, there is a necessity of more sophisticated scheduling policy.

Example 2: Assume there are total 0-199 tracks that are present on each surface of the disk. If request queue is: 68, 172, 4, 178, 130, 40, 118, 136 and initial position of the head is 25. Apply FCFS disk scheduling algorithm and calculate total head movement.

Solution: The requested tracks, in the order received are 68, 172, 4, 178, 130, 40, 118, 136. Initial position of head is 25 tracks.

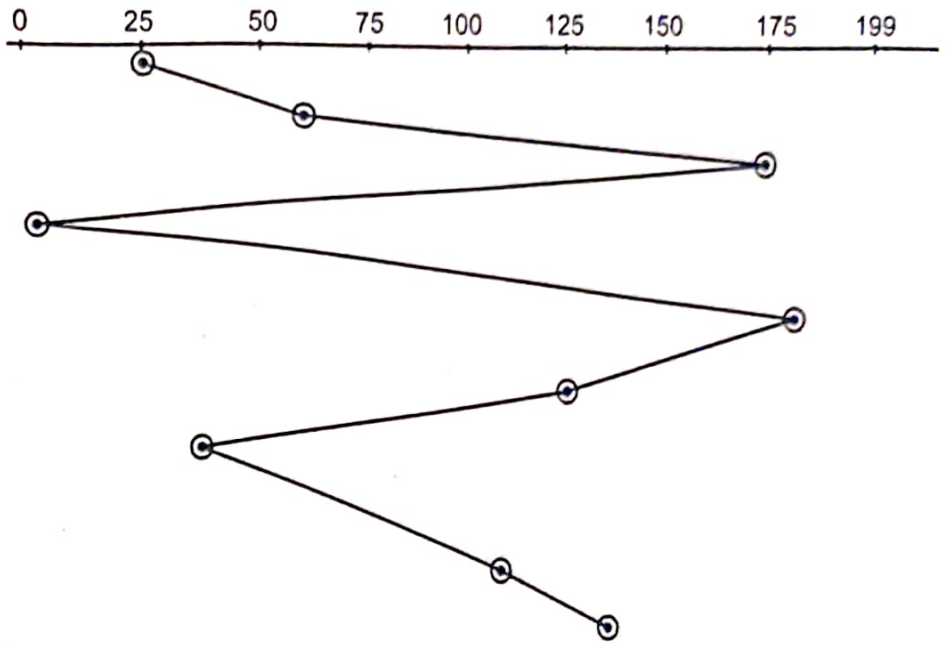


Fig. 9.10

The Total Head Movements:

Next Track Accessed	Number of Track Traversed
25	0
68	43
172	104
04	168
178	174
130	48
40	90
118	78
136	18
Total Head Movements	**723**

Example 3: Suppose the order of request is- (82,170,43,140,24,16,190) And current position of Read/Write head is : 50.

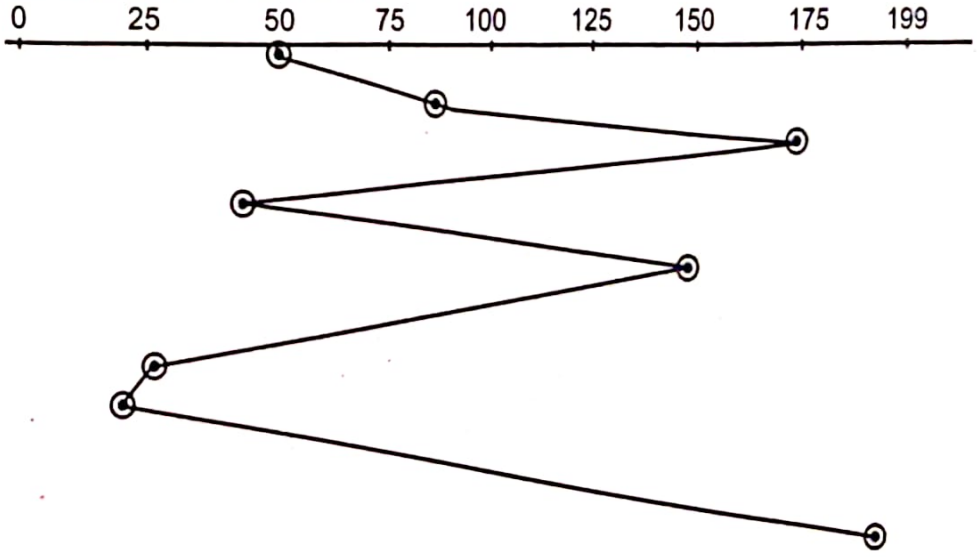


Fig. 9.11

So, total seek time:

= (82 –50) + (170 – 82) + (170 – 43) + (140 – 43) + (140 – 24) + (24 – 16) + (190 – 16)

= 642

Example 4: The request queue is as follows:

28, 136, 15, 185, 50, 197

Number of tracks = 0 to 199.

Starting position or current head position = 134.

Find total head movement by applying FCFS (First Come First Serve) disk scheduling algorithm.

Solution: The requested tracks, in the order received are 28, 136, 15, 185, 50, 197. Initial position of head is at 134 tracks.

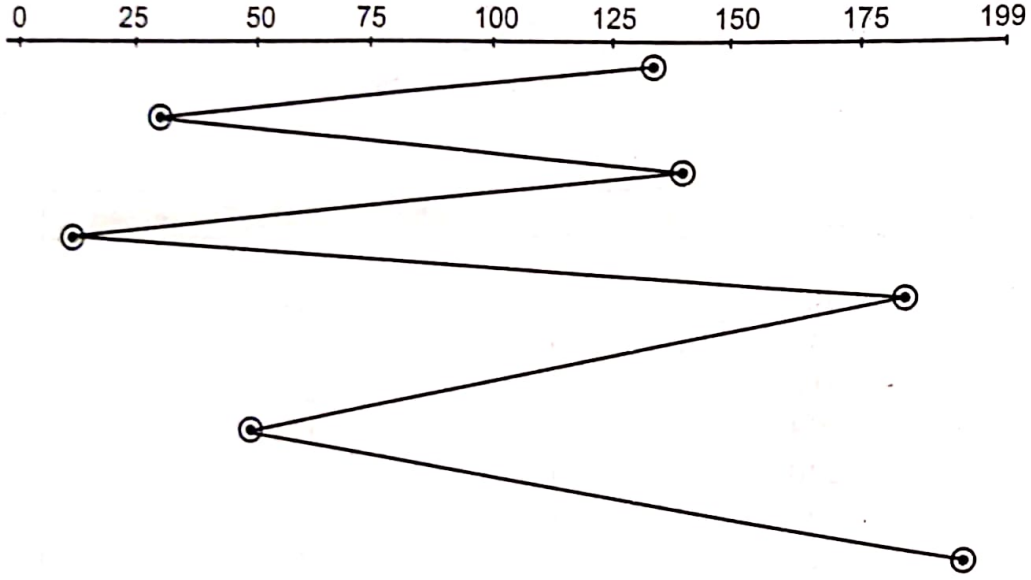


Fig. 9.12: FCFS disk scheduling

Next Tract Accessed	Number of Tracks Traversed
28	106
136	108
15	121
185	170
50	135
197	147
Total Head Movements	**787**

9.5.2 Shortest-Seek-Time-First (SSTF) Algorithm [S-18, 23]

- SSTF selects the request with the minimum seek time from the current head position.
- SSTF scheduling is a form of SJF scheduling; may cause starvation of some requests.

Example 5: Suppose the order of queue is (98, 183, 37, 122, 14, 124, 65, 67) and current position of Read/Write head is : 53

Solution: (53 – 65) + (65 – 67) + (67 – 37) + (37 – 14) + (98 – 14) + (122 – 98) + (124 – 122) + (183 – 124)

Total head movement = 236 cylinders.

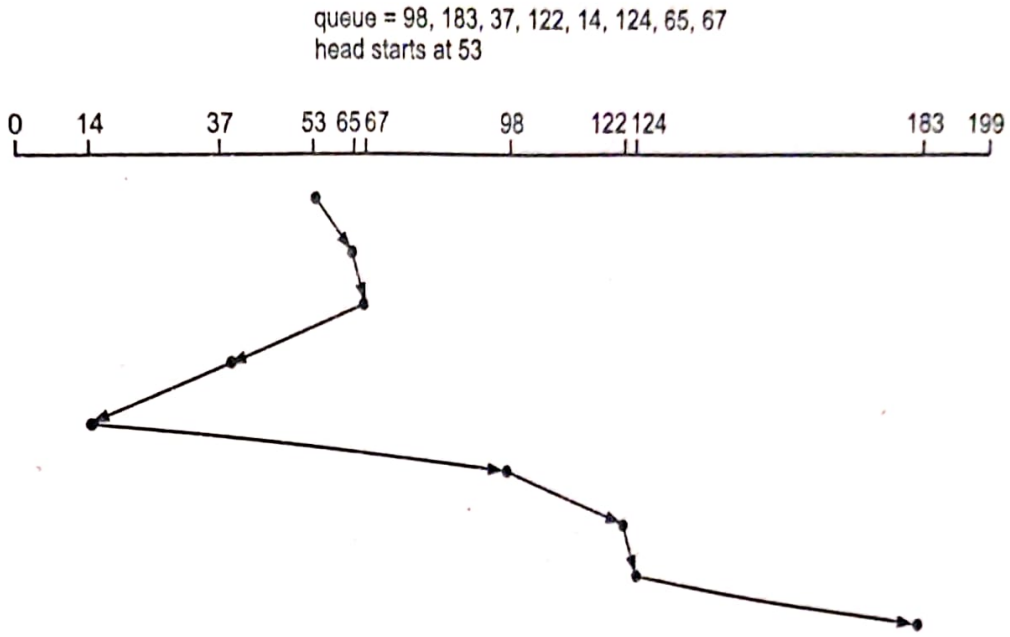


Fig. 9.13: SSTF

Example 6: The request queue is as follows: 15, 27, 137, 18, 150, 65, 194 [W-22; S-23]

Number of tracks = 0 to 199.

Starting position or current head position = 128

Find total head movement by applying SSTF (Shortest Seek Time First) disk scheduling algorithm.

Solution: The requested tracks, in the order received are 15, 27, 137, 18, 150, 65, 194. Initial position of head is at 128 tracks.

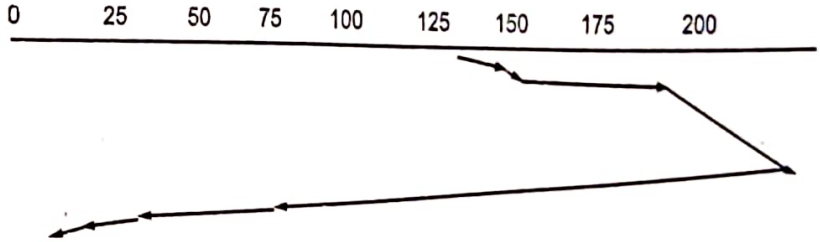


Fig. 9.14

The Total Head Movements:

Next Track Accessed	Number of Track Traversed
128	0
137	9
150	13
194	44
65	129
27	38
18	9
15	3
Total Head Movements	**245**

Example 7: The request queue is as follows:

87, 148, 92, 171, 96, 131, 103, 71

Number of tracks = 0 to 199.

Starting position or current head position = 125

Find total head movement by applying SSTF (Shortest Seek Time First) disk scheduling algorithm.

Solution: The requested tracks, in the order received are 87, 148, 92, 17, 96, 131, 103, 71

Initial position of head is at 125 tracks.

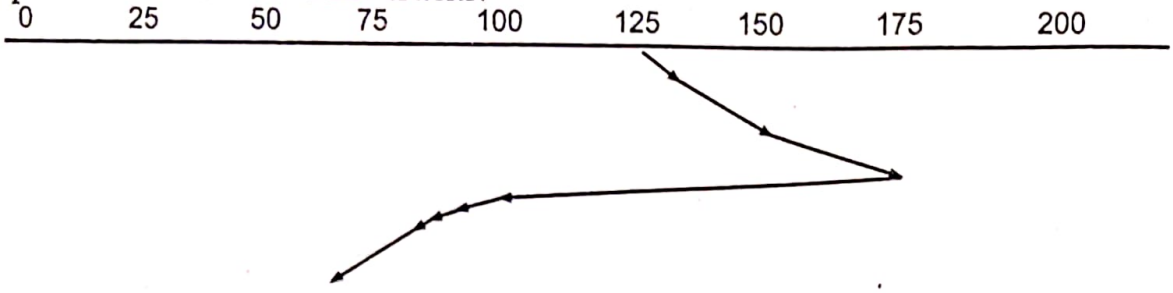


Fig. 9.15

The Total Head Movements:

Next Track Accessed	Number of Track Traversed
125	0
131	6
148	17
171	23
103	68
96	07
92	04
87	05
71	16
Total Head Movements	**146**

In SSTF algorithm, select the disk I/O request that requires the list movement of the disk arm from the current position. Thus, we incur the minimum seek time. Of course, always choosing the minimum seek time does not guarantee that the average seek time over a number of arm movement will be minimum.

This should provide better performance than FIFO. Because the arm can move in the two directions. FIFO algorithm may be used to resolve cases of equal distances.

Example 8: Fig. 9.16 illustrates the disk arm movement with SSTF, on the same example as was used for FIFO.

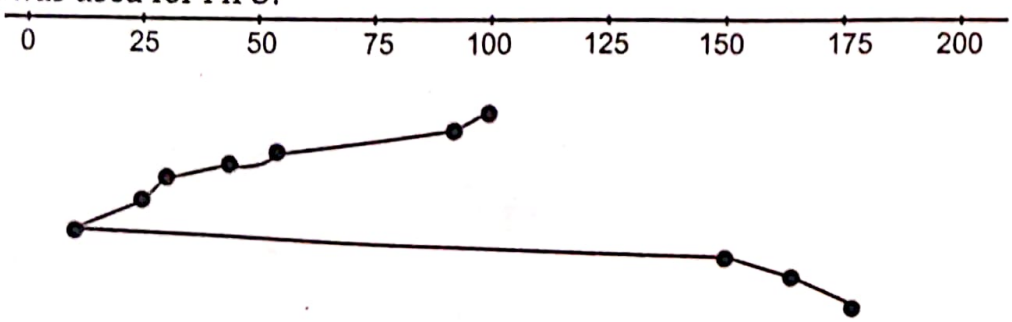


Fig. 9.16: SSTF (Starting at track 100)

Solution: The average seek length is,

Next track accessed	Number of track traversed
90	10
58	32
55	3
39	16
38	1
18	20
150	132
160	10
184	24
Average Seek Length	**27.5**

Unfortunately, SSTF has a problem. Suppose more requests keep coming in while the request of Fig. 9.16 is being processed.

For example, if after going to track 38, a new request for 32 is present, that request will have priority over track 18. If a request for track 25 then comes in, the arm will next go to 25 instead of 18.

With a heavily loaded disk, the arm will tend to stay in middle of disk most of the time, so request at either extreme will have to wait. Requests far from the middle may get poor service.

The goals of minimal response time and fairness are in conflict here. A simple alternative that prevents this sort of starvation is the SCAN algorithm.

9.5.3 SCAN (Elevator Disk Algorithm) [S-19]

- In SCAN, the disk arm starts at one end of the disk and moves toward the other end, servicing requests until it gets to the other end of the disk, where the head movement is reversed and servicing continues.
- Sometimes, called the **Elevator Algorithm.** This algorithm does not give uniform wait time.
- Fig. 9.17 illustrates total head movement of 208 cylinders.
  Total head movement = 208 cylinders.

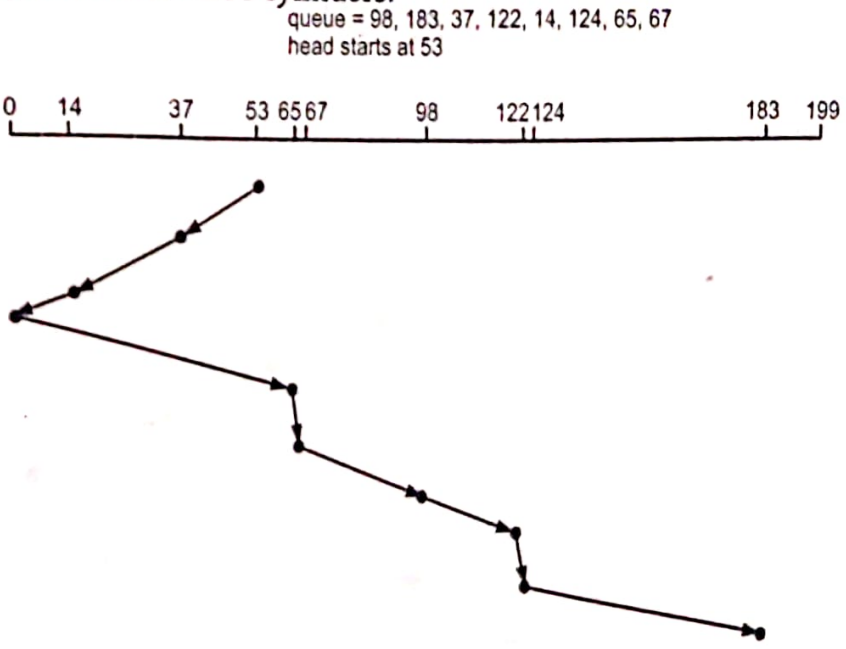


Fig. 9.17: SCAN Disk Algorithm

- In SCAN algorithm, the arm is required to move in one direction only, satisfying all outstanding request in route, until it reaches the last track in that direction or until there are no more requests in that direction.

- The service direction is then reversed and the scan proceed is opposite direction, again picking up all requests in order Fig. 9.18 illustrates the disk arm movement with SCAN, on previous example.

Example 9:

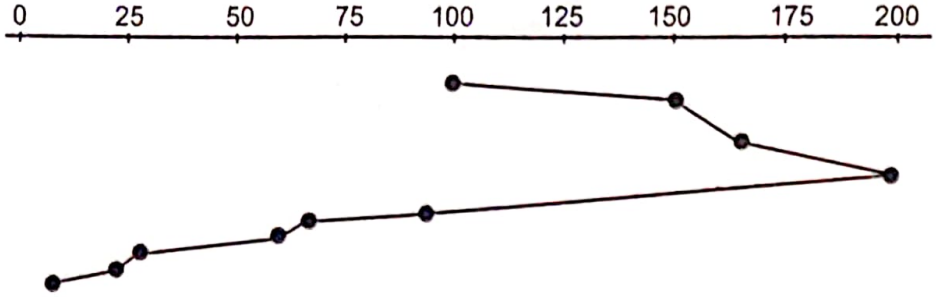


Fig. 9.18: SCAN (Starting at track 100, in the direction of increasing track number)

Solution: The average seek length is,

Next track accessed	Number of track traversed
150	50
160	10
184	24
90	94
58	32
55	3
39	16
38	1
18	20
Average Seek Length	**27.8**

Example 10: The request queue is as follows:

86, 147, 91, 170, 95, 130, 102, 70

Number of Tracks: 0 to 199

Starting position or current head position = 125

Find total head movement by applying SCAN disk scheduling algorithm.

Solution:

The requested tracks, in the order received are 86, 147, 91, 170, 95, 130, 102, 70

Initial position of head is at 125 tracks.

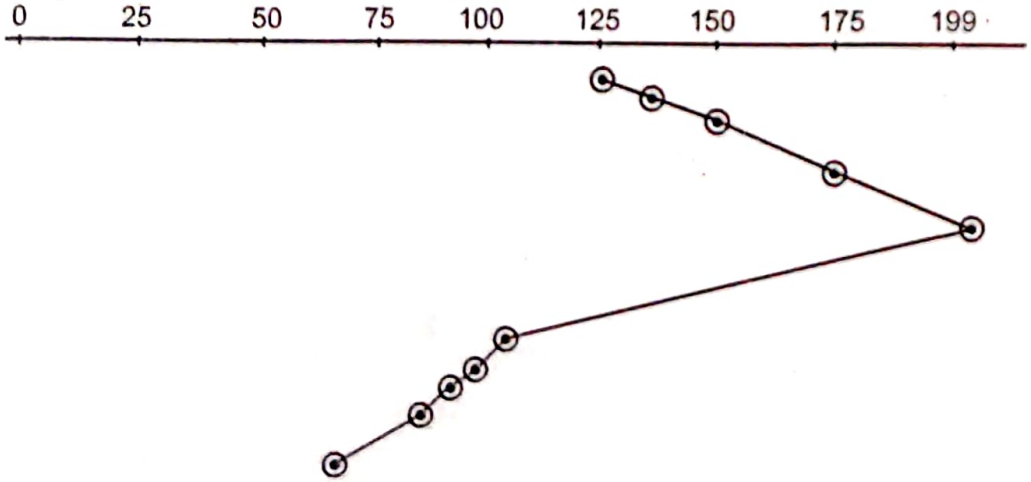


Fig. 9.19: C-SCAN Scheduling

The Total Head Movements:

Next Track Accessed	Number of Track Traversed
125	0
130	5
147	17
170	23
102	68
95	07
91	04
86	05
70	16
Total Head Movements	**145**

SCAN algorithm behaves almost identically with SSTF policy. However, this is a static example in which no new items are added to the queue even when the queue is dynamically changing.

SCAN will be similar to SSTF unless the request pattern is unusual.

SCAN policy favours jobs where requests use for tracks nearest to both innermost and outermost tracks. The problem can be avoided via C-SCAN algorithm.

9.5.4 C-SCAN

- C-SCAN scheduling algorithm provides a more uniform wait time than SCAN algorithm.
- Treats the cylinders as a circular list that wraps around from the last cylinder to the first one.

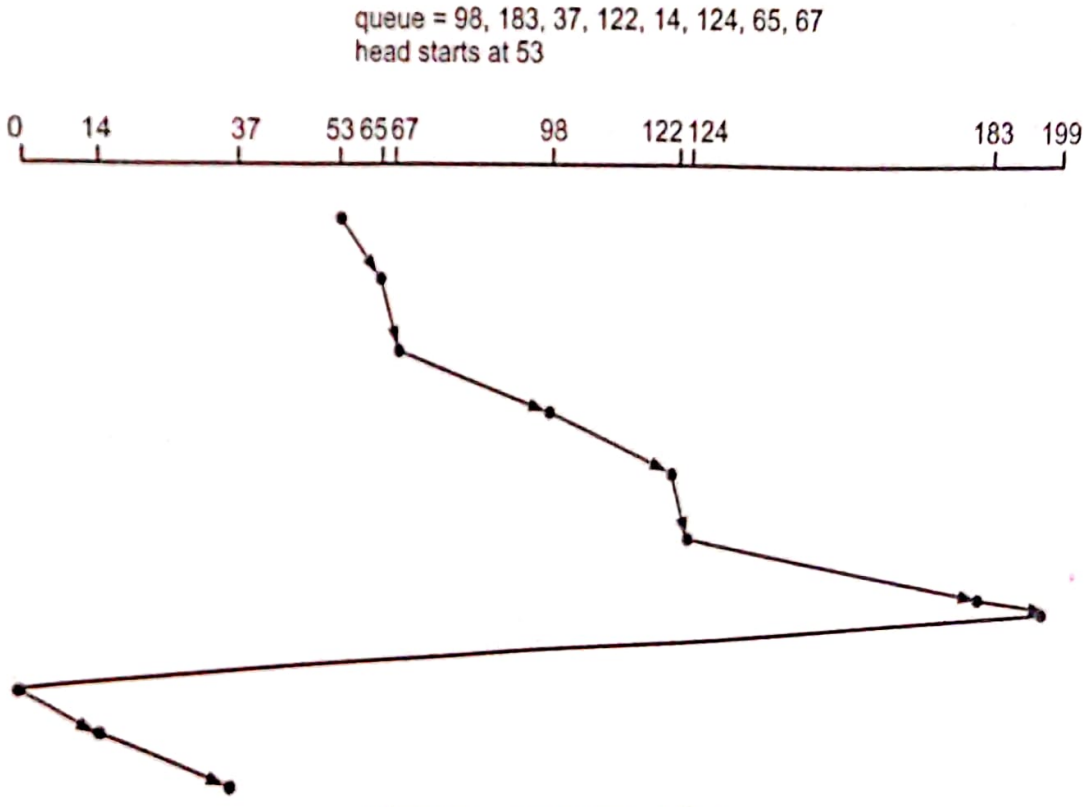


Fig. 9.20: C-SCAN Scheduling

- In C-SCAN algorithm head moves from one end of the disk to the other servicing requests as it goes. When it reaches the other end, however, it immediately returns to the beginning of the disk, without servicing any requests on the return trip.
- The C-SCAN (Circular SCAN) algorithm restricts scanning to one direction only. Thus, when the last track has been visited in one direction, the arm is returned to the opposite end of the disk and the scan begins again.
- This reduces the maximum delay experienced by new requests. Fig. 9.20 illustrates the disk arm movement.

Example 11:

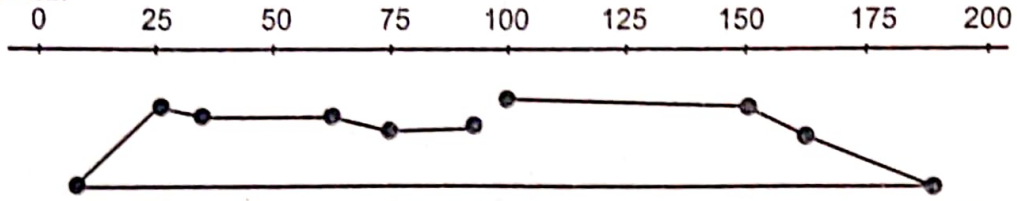


Fig. 9.21: C-SCAN (starting at track 100, in the direction of increasing track number

Solution: The Average seek length is,

Next track accessed	Number of track traversed
150	50
160	10
184	24
18	166
38	20
39	1
55	16
58	3
90	32
Average Seek Length	**35.8**

9.5.5 C-LOOK

- C-LOOK is version of C-SCAN algorithm.
- In C-SCAN algorithm, arm only shows as far as the last request in each direction, then reverses direction immediately, without first going all the way to the end of the disk.
- Fig. 9.22 shows C-LOOK disk scheduling.

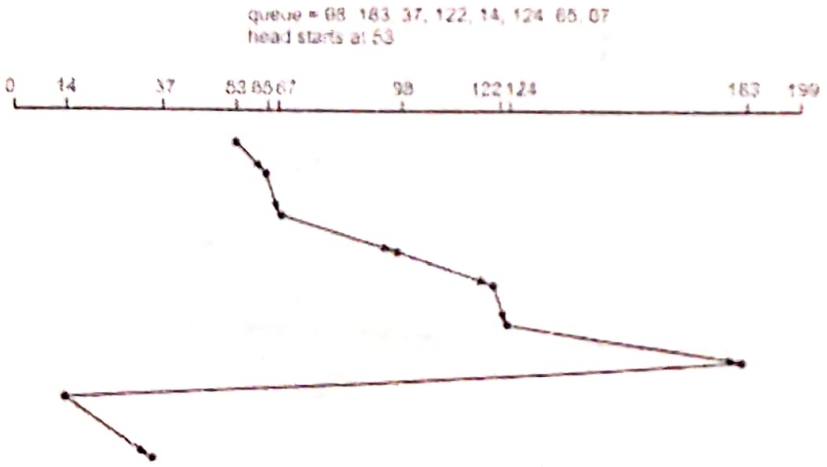


Fig. 9.22: C-LOOK Scheduling

Example 12: Suppose the requests to be addressed are 82,170,43,140,24,16,190. And the Read/Write arm is at 50, and it is also given that the disk arm should move "towards the larger value".

Solution:

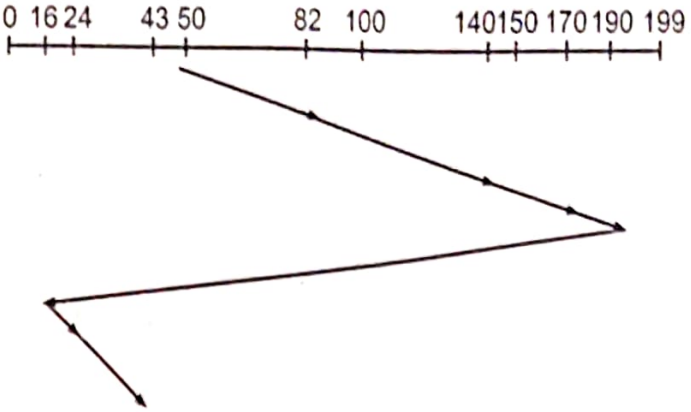


Fig. 9.23

So, the seek time is calculated as:

= (190 – 50) + (190 – 16) + (43 – 16) = 341

Summary

- The basic hardware elements involved in I/O are buses, device controllers, and the devices themselves. The work of moving data between devices and main memory is performed by the CPU as programmed I/O or is offloaded to a DMA controller.
- The system-call interface provided to applications is designed to handle several basic categories of hardware, including block devices, character devices, memory-mapped files, network sockets, and programmed interval timers.
- The kernel's I/O subsystem provides numerous services. These are I/O scheduling, buffering, caching, spooling, device reservation, and error handling.
- Requests for disk I/O are generated by the file system and by the virtual memory system. Each request specifies the address on the disk to be referenced, in the form of a logical block number.
- Disk-scheduling algorithms can improve the effective bandwidth, the average response time, and the variance in response time.
- Algorithms such as SSTF, SCAN, C-SCAN, LOOK, and C-LOOK are designed to make such improvements through strategies for disk-queue ordering. Performance of disk-scheduling algorithms can vary greatly on magnetic disks.
- In contrast, because solid-state disks have no moving parts, performance varies little among algorithms, and quite often a simple FCFS strategy is used.

Check Your Understanding

1. Buffering is done to _____.
   (a) cope with device speed mismatch
   (b) cope with device transfer size mismatch
   (c) maintain copy semantics
   (d) all of the mentioned
2. Caching is _____ spooling.
   (a) same as (b) not the same as
   (c) all of the mentioned (d) none of the mentioned
3. Caching _____.
   (a) holds a copy of the data (b) is fast memory
   (c) holds the only copy of the data (d) holds output for a device

4. Spooling _____.
   (a) holds a copy of the data (b) is fast memory
   (c) holds the only copy of the data (d) holds output for a device
5. The _____ keeps state information about the use of I/O components.
   (a) CPU (b) OS
   (c) kernel (d) shell
6. The kernel data structures include _____.
   (a) process table (b) open file table
   (c) close file table (d) all of the mentioned
7. Windows NT uses a _____ implementation for I/O.
   (a) message – passing (b) draft – passing
   (c) secondary memory (d) cache
8. A _____ is a full duplex connection between a device driver and a user level process.
   (a) Bus (b) I/O operation
   (c) Stream (d) Flow
9. I/O is a _____ in system performance.
   (a) major factor (b) minor factor
   (c) does not matter (d) none of the mentioned
10. If the number of cycles spent busy – waiting is not excessive, then _____.
   (a) interrupt driven I/O is more efficient than programmed I/O
   (b) programmed I/O is more efficient than interrupt driven I/O
   (c) both programmed and interrupt driven I/O are equally efficient
   (d) none of the mentioned
11. Whenever a process needs I/O to or from a disk it issues a _____.
   (a) system call to the CPU (b) system call to the operating system
   (c) a special procedure (d) all of the mentioned
12. If a process needs I/O to or from a disk, and if the drive or controller is busy then _____.
   (a) the request will be placed in the queue of pending requests for that drive
   (b) the request will not be processed and will be ignored completely
   (c) the request will be not be placed
   (d) none of the mentioned
13. Consider a disk queue with requests for I/O to blocks on cylinders.
   98 183 37 122 14 124 65 67
   Considering FCFS (first cum first served) scheduling, the total number of head movements is, if the disk head is initially at 53 is?
   (a) 600 (b) 620
   (c) 630 (d) 640
14. Consider a disk queue with requests for I/O to blocks on cylinders.
   98 183 37 122 14 124 65 67
   Considering SSTF (shortest seek time first) scheduling, the total number of head movements is, if the disk head is initially at 53 is _____.
   (a) 224 (b) 236
   (c) 245 (d) 240
15. Random access in magnetic tapes is _____ compared to magnetic disks.
   (a) fast (b) very fast
   (c) slow (d) very slow

16. Magnetic tape drives can write data at a speed _____ disk drives.
    (a) much lesser than (b) comparable to
    (c) much faster than (d) none of the mentioned
17. On media that use constant linear velocity (CLV), the _____ is uniform.
    (a) density of bits on the disk (b) density of bits per sector
    (c) the density of bits per track (d) none of the mentioned
18. SSTF algorithm, like SJF _____ of some requests.
    (a) may cause starvation (b) will cause starvation
    (c) does not cause starvation (d) causes aging
19. In the _____ algorithm, the disk arm starts at one end of the disk and moves toward the other end, servicing requests till the other end of the disk. At the other end, the direction is reversed and servicing continues.
    (a) LOOK (b) SCAN
    (c) C-SCAN (d) C-LOOK
20. In the _____ algorithm, the disk head moves from one end to the other, servicing requests along the way. When the head reaches the other end, it immediately returns to the beginning of the disk without servicing any requests on the return trip.
    (a) LOOK (b) SCAN
    (c) C-SCAN (d) C-LOOK
21. In the _____ algorithm, the disk arm goes as far as the final request in each direction, then reverses direction immediately without going to the end of the disk.
    (a) LOOK (b) SCAN
    (c) C-SCAN (d) C-LOOK

Answers

1. (d)	2. (b)	3. (a)	4. (c)	5. (c)	6. (b)	7. (a)	8. (c)	9. (a)	10. (b)
11. (b)	12. (a)	13. (d)	14. (b)	15. (d)	16. (b)	17. (c)	18. (a)	19. (b)	20. (c)
21. (a)									

Practice Questions

Q.I Answer the following questions in short:

1. List categories of I/O devices.
2. List techniques for performing I/O.
3. List Disk Scheduling algorithms.
4. What is purpose of Disk Scheduling?
5. Define the term response time.
6. Define the term transfer time.

Q.II Answer the following questions:

1. Explain the term I/O hardware.
2. What is meant by disk scheduling?
3. With the help of diagram describe FCFS disk scheduling algorithm.
4. Describe the term application of I/O interface in detail.
5. With the help of example describe C-LOOK scheduling algorithm.
6. Explain C-SCAN disk scheduling, algorithm with example.
7. Write short note on Kernel I/O subsystem.
8. Compare FCFS and SCAN disk scheduling algorithms.
9. Explain the term SSTF in detail.

10. Assume there are total 200 tracks that are present on each surface of the disk. If request queue is 30, 140, 20, 170, 60, 190 and initial position of the head is 120. Apply FCFS Disk Scheduling and calculate total head movement.
11. Assume there are total 200 tracks that are present on each surface of the disk. If request queue is 70, 120, 10, 180, 90, 50, 100 and initial position of the head is 105. Apply FCFS Disk Scheduling Algorithm and calculate total head movement.
12. Explain DMA with the help of Block Diagram.
13. A disk drive has 540 cylinders numbered 0–539. The drive is currently serving the request at cylinder 54. The queue is in order: 98, 183, 47, 125, 10, 126, 380, 200, 79.
14. Starting from the current head position what is the total distance that the disk arm moves to satisfy all the pending request for the following Disk Scheduling Algorithm?
    (i) FCFS (ii) SCAN

Q.III Define terms:

1. I/O hardware, 2. Buffering, 3. Caching, 4. Spooling, 5. Seek time

Previous Exam Questions

Summer 2018

1. Describe I/O Hardware with its type of I/O devices. **4 M**

Ans. Refer to Section 9.2.

2. The request queue is a follows: **4 M**

87, 148, 92, 171, 96, 131, 103, 71

Number of tracks = 0 to 199

Starting position or current head position = 125. Find total head movement by applying SSTF (Shortest Seek Time First) Disk Scheduling Algorithm.

Ans. Refer to Section 9.5.2.

Winter 2018

1. What do you mean by Seek Time in Disk Scheduling? **2 M**

Ans. Refer to Section 9.5.

2. Assume there are total 200 tracks present on each surface of the disk, if request queue is 57, 60, 41, 20, 92, 162, 152, 40, 186. Initial position of head is at 99 track. Apply FCFS disk scheduling Algorithm and calculate total head movement? **4 M**

Ans. Refer to Section 9.5.1.

Summer 2019

1. Write a note on interrupts. **4 M**

Ans. Refer to Section 9.2.

2. Assume there are total 0-199 tracks that are present on each surface of the disk. If request queue is: 84, 145, 89, 168, 93, 128, 100, 68 and initial position of the head is 125. Apply SCAN disk scheduling algorithm and calculate total head movement. **4 M**

Ans. Refer to Section 9.5.3.

■■■

Solved Question Papers

Summer 2022

Time : $2^1/_2$ **Max. Marks : 70**

Q. 1 Attempt any eight of the following : **[8 × 2 = 16]**

(a) Define the term operating system.

Ans. Refer to Section 1.1.2.

(b) What is meant by multiprocessing system?

Ans. Refer to Section 1.4.3.

(c) What is process?

Ans. Refer to Section 3.1.

(d) Which scheduler controls the degree of Multiprogramming?

Ans. Refer to Section 3.2.2.

(e) Define Burst Time.

Ans. Refer to Section 4.3.

(f) What is semaphores?

Ans. Refer to Section 5.3.

(g) What do you mean by Rollback?

Ans. Refer to Section 6.4.4.2.

(h) What is meant by Address Binding?

Ans. Refer to Section 7.1.2.

(i) List various operation on File.

Ans. Refer to Section 8.1.2.

(j) What do you mean by Seek Time in Disk Scheduling?

Ans. Refer to Section 9.5.

Q. 2 Attempt any four of the following : **[4 × 4 = 16]**

(a) List and explain advantages of Multiprocessor system.

Ans. Refer to Section 1.2.2.

(b) Explain Process Control Block (PCB) in detail with diagram.

Ans. Refer to Section 3.1.2.

(c) Explain different method for recovery from a deadlock.

Ans. Refer to Section 6.4.4.

(d) What is Fragmentation? Explain types of fragmentation in details.

Ans. Refer to Section 7.3.3.

(e) Calculate average turn around time and average waiting time for all set of processes using FCFS algorithm.

Processes	Burst Time	Arrival Time
P_1	5	1
P_2	6	0
P_3	2	2
P_4	4	0

Ans. Refer to Section 4.4.1.

Q. 3 Attempt any four of the following : **[4 × 4 = 16]**

(a) List and explain system calls related to Process and Job control.

Ans. Refer to Section 2.3.

(b) Explain multilevel Feedback queue Algorithm.

Ans. Refer to Section 4.4.4.2.

(c) Describe in detail the 'Dinning Philosopher Problem' Synchronization problem.

Ans. Refer to Section 5.4.3.

(d) Write a note on interrupts.

Ans. Refer to Section 9.2.

(e) Consider the following page reference string:

4, 6, 7, 8, 4, 6, 9, 6, 7, 8, 4, 6, 7, 9.

The number of Frames is 3. Show page trace and calculate page Fault for the following page replacement schemes.
(i) FIFO, (ii) LRU

Ans. Refer to Sections 7.6.4.1 and 7.6.4.3.

Q. 4 Attempt any four of the following: **[4 × 4 = 16]**

(a) What is meant by Free Space Management? Define Bit vector and Grouping.

Ans. Refer to Sections 8.5, 8.5.1 and 8.5.3.

(b) Define the terms :
(i) Logical Address, (ii) Physical Address

Ans. Refer to Section 7.1.3.

(c) Explain Resource Allocation Graph in detail.

Ans. Refer to Section 6.5.2.2.

(d) What are the difference between Preemptive and Non-Preemptive Scheduling.

Ans. Refer to Sections 4.2.3 and 4.2.4.

(e) Assume there are total 0-199 tracks that are present on each surface of the disk. If request queue is 68, 172, 4, 178, 130, 40, 118 and 136 initial position of the head is 25. Apply FCFS disk scheduling algorithm and calculate total head Movement.

Ans. Refer to Section 4.4.1.

Q. 5 Write a short note on any two of the following : **[2 × 3 = 6]**

(a) Write short note on solution for critical section problem.

Ans. Refer to Section 5.2.

(b) Write a short note on Medium-term schedular.

Ans. Refer to Section 3.2.2.

(c) Explain Indexed Allocation briefly.

Ans. Refer to Section 8.4.3.

■■■

Winter 2022

Time : $2^1/_2$ **Max. Marks : 70**

Q.1 Attempt any Eight of the following: **[2 × 8 = 16]**

(a) Define the term operating system.

Ans. Refer to Section 1.1.2.

(b) Define system program.

Ans. Refer to Section 2.11.

(c) Which scheduler controls the degree of multiprogramming?

Ans. Refer to Section 3.2.2.

(d) What is Turn-Around Time?

Ans. Refer to Section 4.3.

(e) What is meant by Deadlock?

Ans. Refer to Section 5.3.4.

(f) What is demand paging?

Ans. Refer to Section 7.6.2.

(g) List any four attributes of files.

Ans. Refer to Section 8.1.1.

(h) What do you mean by Seek Time in Disk Scheduling.

Ans. Refer to Section 9.5.

(i) What does FIFO and MFU stand for?

Ans. Refer to Sections 7.6.4.1 and 7.6.4.5.

(j) Define Rollback?

Ans. Refer to Section 6.4.4.2.

Q.2 Attempt any four of the following: **[4 × 4 = 16]**

(a) List and explain services provided by the operating system.

Ans. Refer to Section 1.3.

(b) Explain Process Control Block (PCB) with diagram.

Ans. Refer to Section 3.1.2.

(c) Explain 'Dining Philosopher' Synchronization problem.

Ans. Refer to Section 5.4.3.

(d) What is Fragmentation? Explain types of its in detail.

Ans. Refer to Section 7.3.3.

(e) Describe I/O Hardware with its type of I/O devices.

Ans. Refer to Section 9.2.

Q.3 Attempt any four of the following: **[4 × 4 = 16]**

(a) Explain various types of system programs.

Ans. Refer to Section 2.11.

(b) Explain Indexed Allocation in detail.

Ans. Refer to Section 8.4.3.

(c) The request queue is as follows:
15, 27, 137, 18, 150, 65, 194.
Number of tracks = 0 to 199.
Starting position or current head position = 128. Find total head movement by applying SSTF (Shortest Seek Time First) disk scheduling algorithm.

Ans. Refer to Example 6, Chapter 9.

(d) List any two types of Multiprocessor.

Ans. Refer to Section 1.2.2.

(e) Consider the following set of processes with length of CPU Burst time and arrival time in milliseconds.

Process	Arrival	Time Burst Time
P_1	0	3
P_2	2	6
P_3	4	4
P_4	6	5
P_5	8	2

Calculate turn around time, waiting time, average time and average turn around time using preemptive SJF scheduling algorithm.

Ans. Refer Example 3, Chapter 4 (Solved Examples).

Q.4 Attempt any Four of the following: **[4 × 4 = 16]**

(a) Consider the following snapshot of the system.

Process	Allocation				Max				Available			
	A	B	C	D	A	B	C	D	A	B	C	D
P_0	0	0	1	2	0	0	1	2	1	5	2	0
P_1	1	0	0	0	1	7	5	0				
P_2	1	3	5	4	2	3	5	6				
P_3	0	6	3	2	0	6	5	2				
P_4	0	0	1	4	0	6	5	6				

Is the system safe? Justify?
If yes give sequence.

Ans. Refer to Example 4, Chapter 6.

(b) Explain different methods for recovery from deadlock?

Ans. Refer to Section 6.4.4.

(c) Consider a reference string 4, 7, 6, 1, 7, 6, 1, 2, 7, 2 the number of frames in the memory is 3. Find out number of page Faults respective to: (i) FIFO, (ii) LRU

Ans. Refer to Example 3, Chapter 7.

(d) Explain advantages and disadvantages of linked allocation methods.

Ans. Refer to Section 8.4.2.

(e) Define the term: (i) Logical Address, (ii) Physical Address

Ans. Refer to Section 7.1.3.

Q.5 Write short note on any Two of the following: **[2 × 3 = 6]**

(a) What is Interrupts.

Ans. Refer to Section 9.2.

(b) What is medium term scheduler.

Ans. Refer to Section 3.2.2.

(c) Explain semaphores and its types in detail.

Ans. Refer to Sections 5.3 and 5.3.3.

■■■

Summer 2023

Time : $2^1/_2$ **Max. Marks : 70**

Q. 1 Attempt any Eight of the following : **[8 × 2 = 16]**

(a) Define 'Least Recently Used' in memory management.

Ans. Refer to Section 7.6.4.3

(b) Define Context Switch?

Ans. Refer to Section 3.2.3.

(c) What is a page frame?

Ans. Refer to Section 7.4.1.

(d) List various properties of the file.

Ans. Refer to Section 8.1.1.

(e) What is 'seek time' in Disk scheduling?

Ans. Refer to Section 9.5

(f) What is compaction?

Ans. Compaction is the process in which the free spaces are collected in the large memory chunk to make some space available for processes.

(g) Define Belady's Anomaly.

Ans. The phenomenon in which increasing the number of page frames results in an increase in the number of page faults for certain memory access patterns is called Belady's Anomaly.

(h) List any four characteristics of operating system.

Ans. Refer to Section 1.1.3.

(i) Define a safe state.

Ans. Refer to Section 6.4.2.1.

(j) What is starvation?

Ans. Refer to Section 4.4.3.

Q. 2 Attempt any Four of the following : **[4 × 4 = 16]**

(a) Explain Operating System Structure.

Ans. Refer to Section 1.5.

(b) What is scheduling? Compare short term scheduler with long term scheduler.

Ans. Refer to Section 4.1.

Sr. No.	Long Term Scheduler	Short Term Scheduler
1.	It is an operating system scheduler that chooses processes from the job queue and loads them to execution in the main memory.	It is an operating system scheduler that chooses the process from the several processes that the processor runs.
2.	It is also referred to as the Job Scheduler.	It is also referred to as a CPU scheduler.
3.	It is slower.	It is faster.
4.	It controls the multiprogramming degree.	It provides less control over the multiprogramming degree.
5.	It selects the process less frequently.	It selects the process more frequently.
6.	It is always present in the Batch OS and can or cannot be present at all in the Time-Sharing OS.	It is present in the Batch OS and is only minimally present in the Time-Sharing OS.
7.	It chooses the processes from the job pool.	It chooses the processes from the ready queue.
8.	It chooses a good process that is a mix-up of input/output bound and CPU bound.	It chooses a new process for a processor quite frequently.

(c) Draw and explain Round Robin Scheduling with the help of an example.

Ans. Refer to Section 4.4.4.

(d) What are Semaphores? Explain the types of Semaphores.

Ans. Refer to Section 5.3.

(e) Draw and explain the Contiguous Memory Allocation.

Ans. Refer to Section 7.3.

Q. 3 Attempt any Four of the following : **[4 × 4 = 16]**

(a) State and explain Critical Section Problem.

Ans. Refer to Section 5.2.

(b) Consider the following set of processes with the length of the CPU burst time given in milliseconds –

Process	Burst Time	Arrival Time
P_1	3	3
P_2	3	6
P_3	4	0
P_4	5	2

(i) Draw Gantt chart using non preemptive Shortest Job First method.

(ii) Calculate average Turnaround time & average Waiting time.

Ans. Refer to Example 6 of Chapter 4.

(c) What is a deadlock? How can deadlock be avoided?

Ans. Refer to Sections 6.1 and 6.4.2.

(d) Explain File System Access Methods.

Ans. Refer to Section 8.2.

(e) Explain Paging in case of memory management.

Ans. Refer to Section 7.4.

Q. 4 Attempt any Four of the following : **[4 × 4 = 16]**

(a) Assume there are a total 200 tracks present on the disk, if the request queue is: 82, 170, 43, 140, 24, 16, 190 and the initial position of the head is 50. Apply Shortest Seek Time First (SSTF) disk scheduling algorithm and calculate total head movement.

Ans. Refer to Example 6 of Chapter 9.

(b) Explain Job Control Block with the help of a diagram.

Ans. Refer to Section 2.3.

(c) What are the characteristics and necessary conditions for a deadlock?

Ans. Refer to Sections 6.2 and 6.3.

(d) Consider the page reference string. 4, 7, 6, 1, 7, 6, 1, 2, 7, 2.
The number of frames in the memory is 3. Initially all frames are empty.
Find out the number of page faults respective to :

(i) Optimal Page Replacement Algorithm

(ii) FIFO Page Replacement Algorithm

(iii) LRU Page Replacement Algorithm

Ans. Refer to Sections 7.6.4.1, 7.6.4.2 and 7.6.4.3.

(e) Explain memory management through Fragmentation with the help of a diagram.

Ans. Refer to Sections 7.1 and 7.3.3.

Q. 5 Write a short note on Two of the following : **[2 × 3 = 6]**

(a) Shortest Seek Time First.

Ans. Refer to Section 9.5.2.

(b) Linked Allocation for File System.

Ans. Refer to Section 8.4.2.

(c) Address binding in case of memory management.

Ans. Refer to Section 7.1.2.

■■■