6

Devices: Scheduling of Disk Requests

6.1 Chapter Objective

The objective of project Devices is to teach students about device I/O and, relatedly, certain aspects of device drivers. One main focus will be the scheduling of disk I/O requests. To meet these objectives, students will be asked to implement the three public classes of the Devices package: Device, IORB, and DiskInterruptHandler. The class Device deals with the scheduling of I/O requests, IORB implements the I/O Request Block data structure, and DiskInterruptHandler constitutes the interrupt handler for I/O devices.

6.2 Overview of I/O Handling

I/O supervisor and I/O Request Block. When the user thread issues a read() or write() system call, the OS assembles an input/output request block (or IORB) and passes the request to the basic I/O supervisor: the portion of the operating system responsible for managing the various I/O devices of the system. The IORB includes information about the thread that issued the call; the buffer page in main memory that contains the data to be written out or into which the data is to be copied from the secondary storage; the disk block to which the buffer data is to be written out or which contains the data to be read in; and the I/O device that is the target of the requested
The I/O supervisor examines the IORB and places it on the \textbf{device queue} of the targeted device. A device queue is nothing more than a queue of waiting-to-be-serviced IORBs, one such queue per I/O device in the system.

\textbf{Disk interrupt handler.} When the device finishes servicing an I/O request, a device interrupt occurs, which is the way by which external devices notify the CPU about completion of an I/O operation. The eventual result of an I/O interrupt is that the appropriate device interrupt handler is called. In OSP 2 the only external devices are disks, so the only device interrupt handler is the disk interrupt handler.

A disk interrupt handler performs a variety of functions, which we will describe in detail in Section 6.5. One of these is to invoke the I/O scheduler, which chooses the IORB to be serviced next, assuming the device queue is non-empty; i.e. contains at least one IORB. Once an IORB has been selected, it is dequeued from the device queue and the device is instructed to process the request. If the device queue is empty, the device simply idles.

\textbf{Disk-scheduling algorithms.} A variety of disk-scheduling policies have been proposed for use by the I/O scheduler. Many of these policies are concerned with performance and QoS (Quality of Service) issues related to the physical characteristics of a disk device. Such a device is typically configured as a number of platters, each of which has an upper and lower surface on which data can be magnetically encoded. A surface consists of a number of concentric tracks each of which is divided into storage regions known as sectors. For most disk drives, a fixed sector size of 512 bytes is used. The block size of a disk is the number of bytes transferred in a single I/O operation, and is usually a multiple of the sector size. The preceding discussion therefore tells us that a disk address consists of a surface number, track number, and sector number.

Each surface has its own disk arm, at the end of which is a read/write head that must be positioned over the appropriate track for an I/O operation to occur. The arms are attached to the disk-drive boom, which moves the arms in unison back and forth over the tracks of the various surfaces. This gives rise to the concept of the disk cylinder: the collection of tracks carved out of 3-space by virtue of having all read/write heads positioned over the same-numbered track on all surfaces.

Disk I/O can be slow compared with say the time it takes the CPU to access main memory due to the electromechanical aspects of disk operation. In particular, having to position the disk arm over the correct track before an I/O can take place is the biggest culprit. The time taken by this movement is called