Like the Ethernet hardware described in Chapter 16, disk hardware typically uses Direct-Memory-Access (DMA) to allow the disk to transfer an entire block before interrupting the processor. Also like the Ethernet driver, a disk driver does not understand or examine the contents of data blocks. Instead, the driver merely treats the entire storage device as an array of data blocks.

### 18.3 Operations A Disk Driver Supports

At the device driver level, a disk consists of fixed-size data blocks that can be accessed randomly using three basic operations:

- **Fetch**: Copy the contents of a specified block from the disk to a buffer in memory.
- **Store**: Copy the contents of a memory buffer to a specified block on the disk.
- **Seek**: Move to a specified block on the disk. The seek operation is only used on electro-mechanical devices (i.e., a magnetic disk) because it provides an optimization that positions the disk head where it will be needed in the future. Thus, when solid state disk technology is used, seek is unimportant.

The block size of a disk is derived from the size of a sector on magnetic disks. The industry has settled on a de facto standard block size of 512 bytes; throughout the chapter, we will assume 512-byte blocks.²

### 18.4 Block Transfer And High-level I/O Functions

Because the hardware only provides block transfer, it makes sense to design an interface in which **read** and **write** operations transfer an entire block. The question becomes how to include a block specification in the existing high-level I/O operations. We might require a programmer to call **seek** to move to a specific block before calling **read** or **write** to access data in the block. Unfortunately, requiring a user to call **seek** before each data transfer is clumsy and error prone. Therefore, to keep the interface simple, we will stretch the usual meaning of arguments to **read** and **write**: instead of interpreting the third argument as a buffer size, we will assume the buffer is large enough to hold a disk block, and use the third argument to specify a block number. For example, the call:

```plaintext
read ( DISK0, buff, 5 )
```

requests the driver to read block five from the disk into memory starting at location `buff`.

²Although modern disks often use a larger underlying block size (e.g., 4K bytes), the hardware presents an interface that uses 512-byte blocks.