启动相应的驱动程序。进程分派也在内核完成某些操作，并且需要再次启动一个用户进程时发生。进程分派的代码是汇编代码，并且和进程调度代码有很大不同。

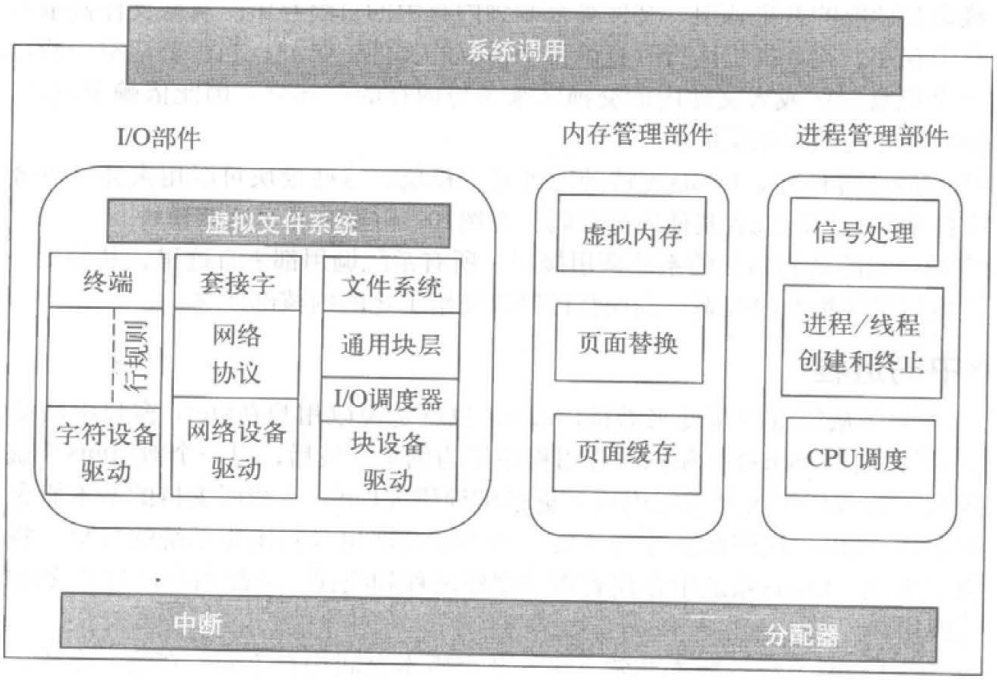

图10-3 Linux内核结构

接下来，我们将内核子系统分为三个主要部件。在图10-3中I/O部件包含所有负责与设备交互以及实现联网和存储的I/O功能的内核部件。在最高层，这些I/O功能全部整合在一个虚拟文件系统层中。也就是说，从顶层来看，对一个文件进行读操作，不论是在内存还是磁盘中，都和从终端输入中读取一个字符是一样的。从底层来看，所有的I/O操作都要通过某一个设备驱动器。所有的Linux驱动程序都可以被分类为字符驱动程序或块驱动程序，两者之间的主要区别是块设备允许查找和随机访问而字符设备不允许。从技术上讲，网络设备实际上是字符设备，不过它们的处理和其他字符设备不太一样，因此为了清晰起见将它们单独分类，如图10-3所示。

在设备驱动程序之上，每个设备类型的内核代码都不一样。字符设备有两种不同的使用方式。有些程序，如可视编辑器vi，emacs等，需要每一个键盘输入。原始的终端（tty）I/O可以实现这种功能。其他程序，比如shell等，是面向行的，因此允许用户在输入回车并将字符串发送给程序之前整行地进行编辑。在这种情况下，由终端流出的字符流需要通过一个所谓的行规则，其中的内容被相应地格式化。

网络软件通常是模块化的，由不同的设备和协议来支持。网络设备的上一个层次负责一种常规程序，确保每一个包被送到正确的设备或协议处理器。大多数Linux系统在内核中包含一个完整的硬件路由器的功能，尽管其性能比硬件路由器的性能差一些。在路由器代码之上的是实际的协议栈，它总是包含IP和TCP协议，也包含一些其他协议。在整个网络之上的是socket接口，它允许程序来为特定的网络和协议创建socket，并为每一个socket返回一个待用的文件描述符。

在磁盘驱动器之上是I/O调度器，它负责排序和分配磁盘读写操作，以尽可能减少磁头的无用移动或者满足一些其他的系统原则为方法。

块设备列的最顶层是文件系统。Linux允许，也确实有多个文件系统同时存在。为了向文件系统的实现隐藏不同硬件设备体系之间的区别，一个通用的块设备层提供了一个可以被所有文件系统使用的抽象。

图10-3的右边是Linux内核的另外两个重要组成部件，它们负责存储和进程管理任务。存储管理任务包括维护虚拟内存到物理内存的映射，维护最近被访问页面的缓存以及实现一个好的页面置换算法，并且根据需要把需要的数据和代码页读入内存中。

进程管理部件的最主要任务是进程的创建和终止。它还包括一个进程调度器，负责选择下一步运行哪个进程或线程。我们将在下一节看到，Linux把进程和线程简单地看作可运行的实体，并使用统一的