系统调用

林祥，PB16020923

通过上一章，我们知道操作系统的设计目的是为了让人们更加方便、高效地使用计算机。因此，人们编写程序的时候，往往不会直接操纵硬件和关注十分细枝末节的方面，而是通过对操作系统提供的所谓的“系统调用（System Call）”进行组织来实现相应的操作。实际上，系统调用是多层次的，每一层次只关心如何组织低层次的系统调用来实现本层次的功能，而不关心低层次的系统调用具体是如何实现的。具体到使用高级语言编写程序的开发人员来说，他们也不直接使用实际的系统调用，而是使用操作系统提供的应用程序接口（Application Programming Interface，简称 API）。API 是一系列适应于具体程序开发的函数，开发人员通过向函数传递参数，来得到相应的返回值或实现某种操作。包括 API 在内的系统调用为开发人员带来的巨大的便利，它向开发人员隐藏了大部分的细节，开发人员只需要知道如何使用它们，这样一来，开发人员可以站在更高、更抽象的层次上来思考问题和编写程序。

系统调用大致可以分为五类：进程控制、文件管理、设备管理、信息维护和通信。

进程控制相关的系统调用包括创建和终止进程、获取进程信息、跟踪进程等等。创建进程的系统调用能决定进程的属性，如优先级和最大允许执行时间。一旦一个进程不再需要或者发生了异常，可以使用系统调用来终止它，同时可以获得内存转储信息来分析和调试程序。通过系统调用还可获得程序的时间表，它具有跟踪功能和定时时间中断，如果中断足够频繁，可以得到程序各部分所用的时间。

文件管理相关的系统调用主要包括创建、读写、删除文件，查看、修改文件属性等操作，更高级的系统调用还可以定位、移动、复制文件。另外，可以通过系统调用创建目录来组织和管理文件。

设备管理相关的系统调用包括对物理设备的识别、访问等。用户程序通过请求操作系统来获得对设备进行访问的权限，一旦获得使用权限，就可通过系统调用进行读写等操作，正如对待文件一样。用户使用完成后，通过系统调用释放它。

信息维护相关的系统调用包括获取当前的时间和日期、获取当前空闲内存/磁盘大小、获取当前用户数以及一些操作系统的其他信息等等，通过系统调用实现用户与系统之间的信息交换。

通信模型包括消息传递模型和共享内存模型。消息传递模型是指进程间通过操作系统的专用程序进行彼此的信息交换，并通过一定的标识符进行识别。一般而言，消息传递模型更容易实现，适合传递少量的信息。共享内存模型通过指定一块公共的内存区域，使两个或多个程序能同时拥有该区域的访问权，这种方法通信速度快且能共享较多的数据，但是建立方式较为复杂。

参考文献:

[1] Abraham Silberschatz. 操作系统概念. 高等教育出版社, 2010.1.