3. 延迟过程调用

Control对象包括线程、中断、定时器、同步、调试等一些原语对象，和两个用来实现DPC和APC的特殊对象。DPC（延迟过程调用）对象是用来减少执行ISR（中断服务例程）所需要的时间，以响应从特定的设备来的中断。

系统硬件为中断指定了硬件优先级。在CPU进行工作时也伴随着一个优先级。CPU只响应比当前更高优先级的中断。通常的优先级是0，包括所有用户态下的优先级。设备中断发生在优先级3或更高，让一个设备中断的ISR以同一优先级的中断来执行是防止其他不重要的中断影响它正在进行的重要中断。

如果ISR执行得太长，提供给低优先级中断的服务将被推迟，可能造成数据丢失或减缓系统的I/O吞吐量。多ISR可以在任何同一时刻处理，每一个后续的ISR是由于产生了更高优先级的中断。

为了减少处理ISR所花费的时间，只有关键的操作才执行，如I/O操作结果的捕捉和设备重置。直到CPU的优先级降低，且没有其他中断服务阻塞，才会进行下一步的中断处理。DPC对象用来表示将要做的工作，ISR调用核心层排列DPC到特定处理器上的DPC队列。如果DPC在队列的第一个位置，内核会登记一个特殊的硬件请求让CPU在优先级2产生中断（NT下称为DISPATCH级别）。当最后一个执行的ISR完成后，处理器的中断级别将回落到低于2，这将解开DPC处理中断。服务于DPC中断的ISR将会处理内核排列好的每一个DPC对象。

利用软中断延迟中断处理是一种行之有效的减少ISR延迟时间的方法。UNIX和其他系统在20世纪70年代开始使用延迟处理，以处理缓慢的硬件和有限的缓冲串行连接终端。ISR负责处理从硬件提取字符并排列它们。在所有高级别的中断处理完成以后，软中断将执行一个低优先级的ISR做字符处理，比如通过向终端发送控制字符来执行一个退格键，以抹去最后一个显示字符并向后移动光标。

在当前的Windows操作系统下，类似的例子是键盘设备。当一个键被敲击以后，键盘ISR从寄存器中读取键值，然后重新使键盘中断，但并不对下一步的按键进行及时处理。相反，它使用一个DPC去排队处理键值，直到所有优先的设备中断已处理完成。

因为DPC在级别2上运行，它们并不干涉ISR设备的执行，在所有排队中的DPC执行完成并且CPU的优先级低于2之前，它们会阻止任何线程的运行。设备驱动和系统本身必须注意不要运行ISR或DPC太长时间。因为在运行它们的时候不能运行线程，ISR或DPC的运行会使系统出现延迟，并且可能在播放音乐时产生不连续，因为拖延了线程对声卡的音乐缓冲区的写操作。DPC另一个通常的用处是运行程序以响应定时器中断。为了避免线程阻塞，要延长运行时间的定时器事件需要向内核维持后台活动的线程工作池做排队请求。这些线程有调度优先级12、13或15。我们会在线程调度部分看到，这些优先级意味着工作项目将会先于大多数线程执行，但是不会打断实时线程。

4. 异步过程调用

另一个特殊的内核控制对象是APC（异步过程调用）对象。APC与DPC的相同之处是它们都是延迟处理系统例行程序，不同之处在于DPC是在特定的CPU上下文中执行，而APC是在一个特定的线程上下文中执行。当处理一个键盘敲击操作时，DPC在哪一个上下文中运行是没有关系的，因为一个DPC仅仅是处理中断的另一部分，中断只需要管理物理设备和执行独立线程操作，例如在内核空间的一个缓冲区记录数据。

当原始中断发生时，DPC例程运行在任何线程的上下文中。它利用I/O系统来报告I/O操作已经完成，I/O系统排列一个APC在线程的上下文中运行从而做出原始的I/O请求，在这里它可以访问处理输入的线程的用户态地址空间。

在下一个合适的时间，内核层会将APC移交给线程而且调度线程运行。一个APC被设计成看上去像一个非预期的程序调用，有些类似于UNIX中的信号处理程序。不过在内核态下，内核态的APC为了完成I/O操作，而在完成初始化I/O操作的线程的上下文中执行。这使APC既可以访问内核态的缓冲区，又可以访问用户态下，属于包含线程的进程的地址空间。一个APC在什么时候被移交，取决于线程已经在做什么，以及系统的类型是什么。在一个多处理器系统中，甚至是在DPC完成运行之前，接收APC的线程才可以开始执行。

用户态下的APC也可以用来把用户态的I/O操作已经完成的信息，通知给初始化I/O操作的线程。但