Linux 内核中的 `do_fork()` 和 `do_exit()`

> 2021240205班 严晏来 2021902228

`do_fork()` 函数：

功能：

`do_fork()` 是 Linux 内核中负责创建新进程的关键函数。它是通过 `fork()` 系统调用触发的，用于复制当前进程的执行上下文，包括代码段、数据段、文件描述符等，以创建一个新的进程。

主要步骤：

1. **创建新的进程描述符 (`task_struct`)：** 在内核中，每个进程都有一个进程描述符，`do_fork()` 负责创建新进程的描述符，并复制父进程的信息。
2. **分配新的地址空间：** 为新进程分配一个独立的地址空间，这包括代码、数据和栈等。
3. **复制文件描述符表：** 新进程与父进程共享打开的文件。`do_fork()` 负责复制父进程的文件描述符表，以确保文件共享。
4. **设置新进程状态和标志：** 设置新进程的状态，如就绪态、运行态等，并设置相应的标志位。
5. **调度新进程：** 调度器将新进程添加到就绪队列，以便在适当的时机执行。

代码位置：

`do_fork()` 的代码主要位于 `kernel/fork.c` 文件中。

`do_exit()` 函数：

功能：

`do_exit()` 是 Linux 内核中用于处理进程退出的函数。它执行与进程终止相关的清理工作，包括释放资源、关闭文件、通知父进程等。

主要步骤：

1. **释放进程的资源：** 包括文件描述符、内存等。
2. **通知父进程：** 通过发送信号（`SIGCHLD`）通知父进程，表示当前进程已经终止。
3. **释放进程描述符 (`task_struct`)：** 释放当前进程的描述符。
4. **调度新的任务：** 如果父进程正在等待子进程退出，调度器会将 CPU 时间分配给等待的父进程。

代码位置：

`do_exit()` 的代码主要位于 `kernel/exit.c` 文件中。

总结：

`do_fork()` 和 `do_exit()` 是 Linux 内核中用于处理进程创建和退出的两个重要函数。它们负责管理进程的生命周期，包括创建新的进程并在进程退出时进行清理。这两个函数的实现是 Linux 内核中进程管理的核心组成部分，对于操作系统的稳定性和性能至关重要。在深入理解这两个函数的基础上，我们能更好地理解 Linux 操作系统的工作原理。