深入浅出Linux设备驱动异步通知介绍 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/221/2021_2022__E6_B7_B1_E5_85_A5_E6_B5_85_E5_c103_221610.htm 结合阻塞与非阻塞访问、poll函数可以较好地解决设备的读写，但是如果有了异步通知就更方便了。异步通知的意思是：一旦设备就绪，则主动通知应用程序，这样应用程序根本就不需要查询设备状态，这一点非常类似于硬件上"中断"地概念，比较准确的称谓是"信号驱动(SIGIO)的异步I/O"。 我们先来看一个使用信号驱动的例子，它通过signal(SIGIO, input_handler)对STDIN_FILENO启动信号机制，输入可获得时input_handler被调用，其源代码如下： #include #include #include #include #include #include #define MAX_LEN 100void input_handler(int num){ char data[MAX_LEN]. int len. //读取并输出STDIN_FILENO上的输入 len = read(STDIN_FILENO, &amp.data, MAX_LEN). data[len] = 0. printf("input available:%s\n", data).}main(){ int oflags. //启动信号驱动机制 signal(SIGIO, input_handler). fcntl(STDIN_FILENO, F_SETOWN, getpid()). oflags = fcntl(STDIN_FILENO, F_GETFL). fcntl(STDIN_FILENO, F_SETFL, oflags | FASYNC). //最后进入一个死循环，程序什么都不干了，只有信号能激发input_handler的运行 //如果程序中没有这个死循环，会立即执行完毕 while (1). }为了使设备支持该机制，我们需要在驱动程序中实现fasync()函数，并在write()函数中当数据被写入时，调用kill_fasync()函数激发一个信号，此部分工作留给读者来完成。 100Test 下载频道开通，各类考试题目直接下载。详