Linux系统进程间隔定时器Itimer(上)（2） PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/144/2021_2022_Linux_E7_B3_BB_E7_BB_c103_144902.htm 7．7．1 数据结构itimerval 虽然，在内核中间隔定时器的间隔计数器是以时钟滴答次数为单位，但是让用户以时钟滴答为单位来指定间隔定时器的间隔计数器的初值显然是不太方便的，因为用户习惯的时间单位是秒、毫秒或微秒等。所以Linux定义了数据结构itimerval来让用户以秒或微秒为单位指定间隔定时器的时间间隔值。其定义如下（include/linux/time.h）： struct itimerval { struct timeval it_interval. /* timer interval */ struct timeval it_value. /* current value */ }.其中，it_interval成员表示间隔计数器的初始值，而it_value成员表示间隔计数器的当前值。这两个成员都是timeval结构类型的变量，因此其精度可以达到微秒级。 timeval与jiffies之间的相互转换 由于间隔定时器的间隔计数器的内部表示方式与外部表现方式互不相同，因此有必要实现以微秒为单位的timeval结构和为时钟滴答次数单位的jiffies之间的相互转换。为此，Linux在kernel/itimer.c中实现了两个函数实现二者的互相转换tvtojiffies()函数和jiffiestotv()函数。它们的源码如下： static unsigned long tvtojiffies(struct timeval *value) { unsigned long sec = (unsigned) value->tv_sec. unsigned long usec = (unsigned) value->tv_usec. if (sec > (ULONG_MAX / HZ)) return ULONG_MAX. usec = 1000000 / HZ - 1. usec /= 1000000 / HZ. return HZ*sec usec. } static void jiffiestotv(unsigned long jiffies, struct timeval *value) { value->tv_usec = (jiffies % HZ) * (1000000 / HZ). value->tv_sec = jiffies / HZ. }7．7．2 真实间隔定时

器ITIMER_REAL的底层运行机制 间隔定时器ITIMER_VIRT和ITIMER_PROF的底层运行机制是分别通过函数do_it_virt（）函数和do_it_prof（）函数来实现的，这里就不再重述（可以参见7.4.3节）。 由于间隔定时器ITIMER_REAL本质上与内核动态定时器并无区别。因此内核实际上是通过内核动态定时器来实现进程的ITIMER_REAL间隔定时器的。为此，task_struct结构中专门设立一个timer_list结构类型的成员变量real_timer。动态定时器real_timer的函数指针function总是被task_struct结构的初始化宏INIT_TASK设置为指向函数it_real_fn()。如下所示（include/linux/sched.h）：#define INIT_TASK(tsk) \…… real_timer：{function：it_real_fn \} \…… }而real_timer链表元素list和data成员总是被进程创建时分别初始化为空和进程task_struct结构的地址，如下所示（kernel/fork.c）：int do_fork(……) {…… p->it_real_value = p->it_virt_value = p->it_prof_value = 0. p->it_real_incr = p->it_virt_incr = p->it_prof_incr = 0. init_timer(&amp.p->real_timer). p->real_timer.data = (unsigned long)p. …… } 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com