Linux操作系统内核对RTC的编程详解（5） 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_E6_93_8D_E4_BD_c103_144831.htm (2)从第二个for循环退出后，RTC的Update Cycle已经结束。此时我们就已经把当前时间逻辑定准在RTC的当前一秒时间间隔内。也就是说，这是我们就可以开始从RTC寄存器中读取当前时间值。但是要注意，读操作应该保证在244us内完成（准确地说，读操作要在RTC的下一个更新周期开始之前完成，244us的限制是过分偏执的：－）。所以，get_cmos_time()函数接下来通过CMOS_READ()宏从RTC中依次读取秒、分钟、小时、日期、月份和年分。这里的do{}while(sec!=CMOS_READ(RTC_SECOND))循环就是用来确保上述6个读操作必须在下一个Update Cycle开始之前完成。 （3）接下来判定时间的数据格式，PC机中一般总是使用BCD格式的时间，因此需要通过BCD_TO_BIN()宏把BCD格式转换为二进制格式。 （4）接下来对年分进行修正，以将年份转换为“ 19XX ”的格式，如果是1970以前的年份，则将其加上100。 （5）最后调用mktime()函数将当前时间与日期转换为相对于1970－01－01 00：00：00的秒数值，并将其作为函数返回值返回。 函数mktime()定义在include/linux/time.h头文件中，它用来根据Gauss算法将以year/mon/day/hour/min/sec（如1980－12－31 23：59：59）格式表示的时间转换为相对于1970－01－01 00：00：00这个UNIX时间基准以来的相对秒数。其源码如下： static inline unsigned long mktime (unsigned int year, unsigned int mon,

unsigned int day, unsigned int hour, unsigned int min, unsigned int sec) { if (0 >= (int) (mon -= 2)) { /* 1..12 -> 11,12,1..10 */ mon = 12. /* Puts Feb last since it has leap day */ year -= 1. } return (((((unsigned long) (year/4 - year/100 year/400 367*mon/12 day) year*365 - 719499)*24 hour /* now have hours */)*60 min /* now have minutes */)*60 sec. /* finally seconds */ }（3）set_rtc_mmss()函数 该函数用来更新RTC中的时间，它仅有一个参数nowtime，是以秒数表示的当前时间，其源码如下： static int set_rtc_mmss(unsigned long nowtime) { int retval = 0. int real_seconds, real_minutes, cmos_minutes. unsigned char save_control, save_freq_0select. /* gets recalled with irq locally disabled */ spin_lock(amp. RTC_DM_BINARY) || RTC_ALWAYS_BCD) BCD_TO_BIN(cmos_minutes). /* * since were only adjusting minutes and seconds, * dont interfere with hour overflow. This avoids * messing with unknown time zones but requires your * RTC not to be off by more than 15 minutes */ real_seconds = nowtime % 60. real_minutes = nowtime / 60. if (((abs(real_minutes - cmos_minutes) 15)/30) &amp. 1) real_minutes = 30. /* correct for half hour time zone */ real_minutes %= 60. if (abs(real_minutes - cmos_minutes) 100