Linux操作系统内核对RTC的编程详解（1） 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_144838.htm Linux内核对RTC的编程 MC146818 RTC芯片（或其他兼容芯片，如DS12887）可以在IRQ8上产生周期性的中断，中断的频率在2HZ～8192HZ之间。与MC146818 RTC对应的设备驱动程序实现 在include/linux/rtc.h和drivers／char/rtc.c文件中，对应的设备文件是／dev/rtc（major=10,minor=135，只读字符设备）。因此用户进程可以通过对她进行编程以使得当RTC到达某个特定的时间值时激活IRQ8线，从而将RTC当作一个闹钟来用。而Linux内核对RTC的唯一用途就是把RTC用作“ 离线 ”或“ 后台 ”的时间与日期维护器。当Linux内核启动时，它从RTC中读取时间与日期的基准值。然后再运行期间内核就完全抛开RTC，从而以软件的形式维护系统的当前时间与日期，并在需要时将时间回写到RTC芯片中。 Linux 在include/linux/mc146818rtc.h和include/asm-i386/mc146818rtc.h头文件中分别定义了mc146818 RTC芯片各寄存器的含义以及RTC芯片在i386平台上的I/O端口操作。而通用的RTC接口则声明在include/linux/rtc.h头文件中。 7．2．1 RTC芯片的I/O端口操作 Linux在include/asm-i386/mc146818rtc.h头文件中定义了RTC芯片的I/O端口操作。端口0x70被称为“ RTC端口0 ”，端口0x71被称为“ RTC端口1 ”，如下所示： #ifndef RTC_PORT #define RTC_PORT(x) (0x70 (x)) #define RTC_ALWAYS_BCD 1 /* RTC operates in binary mode */ #endif 显然，RTC_PORT(0)就是指端口0x70，RTC_PORT(1)就是

指I/O端口0x71。 端口0x70被用作RTC芯片内部寄存器的地址索引端口，而端口0x71则被用作RTC芯片内部寄存器的数据端口。再读写一个RTC寄存器之前，必须先把该寄存器在RTC芯片内部的地址索引值写到端口0x70中。根据这一点，读写一个RTC寄存器的宏定义CMOS_READ()和CMOS_WRITE()如下： #define CMOS_READ(addr) ({ \ outb_p((addr),RTC_PORT(0)). \ inb_p(RTC_PORT(1)). \ }) #define CMOS_WRITE(val, addr) ({ \ outb_p((addr),RTC_PORT(0)). \ outb_p((val),RTC_PORT(1)). \ }) #define RTC_IRQ 8在上述宏定义中，参数addr是RTC寄存器在芯片内部的地址值，取值范围是0x00~0x3F，参数val是待写入寄存器的值。宏RTC_IRQ是指RTC芯片所连接的中断请求输入线号，通常是8。 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