1. 如何编写一个LINUX驱动？
2. Linux设备中字符设备与块设备有什么主要的区别？请分别列举一些实际的设备说出它们是属于哪一类设备。

字符设备：字符设备是个能够像字节流（类似文件）一样被访问的设备，由字符设备驱动程序来实现这种特性。字符设备驱动程序通常至少实现open,close,read和write系统调用。字符终端、串口、鼠标、键盘、摄像头、声卡和显卡等就是典型的字符设备。  
 块设备：和字符设备类似，块设备也是通过/dev目录下的文件系统节点来访问。块设备上能够容纳文件系统，如：u盘，SD卡，磁盘等。  
字符设备和块设备的区别仅仅在于内核内部管理数据的方式，也就是内核及驱动程序之间的软件接口，而这些不同对用户来讲是透明的。

1. 查看驱动模块中打印信息应该使用什么命令？如何查看内核中已有的字符设备的信息？如何查看正在使用的有哪些中断号？

1) 查看驱动模块中打印信息的命令：dmesg  
 2) 查看字符设备信息可以用lsmod 和modprobe，lsmod可以查看模块的依赖关系，modprobe在加载模块时会加载其他依赖的模块。  
 3）显示当前使用的中断号cat /proc/interrupt

1. 请简述主设备号和次设备号的用途。如果执行mknod chartest c 4 64，创建chartest

使用的是那一类设备驱动程序。

1）主设备号：主设备号标识设备对应的驱动程序。虽然现代的linux内核允许多个驱动程序共享主设备号，但我们看待的大多数设备仍然按照“一个主设备对应一个驱动程序”的原则组织。  
次设备号：次设备号由内核使用，用于正确确定设备文件所指的设备。依赖于驱动程序的编写方式，我们可以通过次设备号获得一个指向内核设备的直接指针，也可将此设备号当作设备本地数组的索引。  
2）chartest 由驱动程序4管理，该文件所指的设备是64号设备。（感觉类似于串口终端或者字符设备终端）。

1. 设备驱动程序中如何注册一个字符设备？分别解释一下它的几个参数的含义。

注册一个字符设备驱动有两种方法：  
1） void cdev\_init(struct cdev \*cdev, struct file\_operations \*fops)  
该注册函数可以将cdev结构嵌入到自己的设备特定的结构中。cdev是一个指向结构体cdev的指针，而fops是指向一个类似于file\_operations结构（可以是file\_operations结构，但不限于该结构）的指针.  
2） int register\_chrdev(unsigned int major, const char \*namem , struct file)operations \*fopen);  
该注册函数是早期的注册函数，major是设备的主设备号，name是驱动程序的名称，而fops是默认的file\_operations结构（这是只限 于file\_operations结构）。

1. 请简述中断于DMA的区别。Linux设备驱动程序中，使用哪个函数注册和注销中断处理程序？

1）MDA不需CPU参与而中断是需要CPU参与的。  
2）中断注册函数和中断注销函数  
注册中断：  
int request\_irq(unsigned int irq, irqreturn\_t (\*handler) (int, void \*, struct pt\_regs \*), unsigned long flags, const char \*dev\_name, void \*dev\_id);  
注销中断；  
Void free\_irq(unsigned int irq, void \*dev\_id);  
释放中断和中断信号线

1. 中断和轮询哪个效率高？怎样决定是采用中断方式还是采用轮询方式去实现驱动？

中断是CPU处于被中状态下来接受设备的信号，而轮询是CPU主动去查询该设备是否有请求。如果是请求设备是一个频繁请求cpu的设备，或者有大量数据请求的网络设备，那么轮询的效率是比中断高。如果是一般设备，并且该设备请求cpu的频率比较底，则用中断效率要高一些。

1. 驱动中操作物理绝对地址为什么要先ioremap?

因为内核没有办法直接访问物理内存地址，必须先通过ioremap获得对应的虚拟地址。

1. 设备驱动模型三个重要成员是？platfoem总线的匹配规则是？在具体应用上要不要先注册驱动再注册设备？有先后顺序没？

Linux设备模型中三个很重要的概念就是总线,设备,驱动.即bus,device,driver。platform总线是内核注册好的用于管理设备及驱动的一种模式。总线实现好了匹配规则，内核对于往platform总线上注册的设备  
及驱动对应的类型做了抽象。

1. insmod 一个驱动模块，会执行模块中的哪个函数？rmmod呢？这两个函数在设计上要注意哪些？遇到过卸载驱动出现异常没？是什么问题引起的？

insmod调用init函数，rmmod调用exit函数。

注意在init函数中申请的资源在exit函数中要释放，包括存储，ioremap，定时器，工作队列等等。也就是一个模块注册进内核，退出内核时要清理所带来的影响，带走一切不留下一点痕迹。

卸载模块时曾出现卸载失败的情形，原因是存在进程正在使用模块，检查代码后发现产生了死锁的问题。

1. 原子操作你怎么理解？为了实现一个互斥，自己定义一个变量作为标记来作为一个资源只有一个使用者行不行？

所谓原子操作，就是该操作绝不会在执行完毕前被任何其他任务或事件打断，也就说，它的最小的执行单位，不可能有比它更小的执行单位。定义一个变量,比如 int flag =0;  
   if(flag == 0)  
  {  
       flag = 1;  
       操作临界区；  
      flag = 0;  
   }

这样应该不能起到互斥的作用。因为另外一个进程根本没有flag这个变量，自然不用检查就能进入临界区。充当互斥的变量应该是访问进程都可见的

1. 自旋锁和信号量在互斥使用时需要注意哪些？在中断服务程序里面的互斥是使用自旋锁还是信号量？还是两者都能用？为什么？

使用自旋锁的进程不能睡眠，使用信号量的进程可以睡眠。中断服务例程中的互斥使用的是自旋锁，原因是在中断处理例程中，硬中断是关闭的，这样会丢失可能到来的中断。

1. 写一个中断服务需要注意哪些？如果中断产生之后要做比较多的事情你是怎么做的？

一、中断服务程序要注意快进快出，在中断服务程序里面尽量快速采集信息，包括硬件信息，然后推出中断，要做其它事情可以使用工作队列或者tasklet方式。也就是中断上半部和下半部。  
 第二：中断服务程序中不能有阻塞操作。       
 第三：中断服务程序注意返回值，要用操作系统定义的宏做为返回值，而不是自己定义的OK，FAIL之类的。

1. 字符型驱动设备你是怎么创建设备文件的，就是/dev/下面的设备文件，供上层应用程序打开使用的文件？

mknod命令结合设备的主设备号和次设备号，可创建一个设备文件。

1. linux中内核空间及用户空间的区别？用户空间与内核通信方式有哪些？

内核空间中存放的是内核代码和数据，而进程的用户空间中存放的是用户程的代码和数据。

1.使用API .2.使用proc文件系统 3.使用sysfs文件系统+kobject 4.netlink 5.文件. 6.使用mmap系统调用：可以将内核空间的地址映射到用户空间。 7.信号：从内核空间向进程发送信号。

1. linux中内存划分及如何使用？虚拟地址及物理地址的概念及彼此之间的转化，高端内存概念？ 4G的进程地址空间被人为的分为两个部分——用户空间与内核空间。用户空间从0到3G（0xC0000000），内核空间占据3G到4G。用户进程通常情况下只能访问用户空间的虚拟地址，不能访问内核空间虚拟地址。只有用户进程进行系统调用（代表用户进程在内核态执行）等时刻可以访问到内核空间。

**虚拟地址,**即逻辑地址，是指由程序产生的与段相关的偏移地址部分。物理地址 (physical address): 放在寻址总线上的地址。

地址空间大于1G的内存区域称之为高端内存

1. linux中中断的实现机制，tasklet与workqueue的区别及底层实现区别？为什么要区分上半部和下半部？

内核把中断处理分为两部分：上半部（tophalf）和下半部（bottomhalf），上半部（就是中断服务程序）内核立即执行，而下半部（就是一些内核函数）留着稍后处理。下半部运行时是允许中断请求的，而上半部运行时是关中断的，这是二者之间的主要区别。

1. linux中断的响应执行流程？中断的申请及何时执行(何时执行中断处理函数)？
2. linux中的同步机制？spinlock与信号量的区别？

Linux中的同步机制(一)--Futex Futex是一种用户态和内核态混合机制，所以需要两个部分合作完成，linux上提供了sys\_futex系统调用，对进程竞争情况下的同步处理提供支持。

# Linux中的线程同步机制(二)--In Glibc 在linux中进行多线程开发，同步是不可回避的一个问题。在POSIX标准中定义了三种线程同步机制: Mutexes(互斥量), Condition Variables(条件变量)和POSIX Semaphores(信号量)。

# Linux中的线程同步机制(三)--Practice

自旋锁(spin lock)是一个典型的对临界资源的互斥手段,Linux中的信号量是一种睡眠锁。自旋锁不会引起调用者睡眠，如果自旋锁已经被别的执行单元保持，调用者就一直循环查看是否该自旋锁的保持者已经释放了锁，"自旋"就是"在原地打转"。而信号量则引起调用者睡眠，它把进程从运行队列上拖出去，除非获得锁。

1. linux系统实现原子操作有哪些方法？

方法1：直接点击wmware的关机按钮  
方法2：进入操作界面点击关机按钮  
方法3：打开终端，以root用户身份执行shutdown命令

1. linux内核里面，内存申请有哪几个函数，各自的区别？

Kmalloc()   \_\_get\_free\_page()  mempool\_create()

1. 关键字volatile有什么含意 并给出三个不同的例子。

优化器在用到这个变量时必须每次都小心地重新读取这个变量的值，而不是使用保存在寄存器里的备份。下面是volatile变量的几个例子：  
　　1). 并行设备的硬件寄存器（如：[状态寄存器](https://www.baidu.com/s?wd=%E7%8A%B6%E6%80%81%E5%AF%84%E5%AD%98%E5%99%A8&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YLmhmLmvcdmHIWuAckPWc30ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3EnWfYn1fsnHf3" \t "http://zhidao.baidu.com/question/_blank)）  
　　2). 一个中断服务子程序中会访问到的非自动变量(Non-automatic variables)  
　　3). 多线程应用中被几个任务共享的变量

1. kmalloc和vmalloc的区别

1、kmalloc保证分配的内存在物理上是连续的,vmalloc保证的是在虚拟地址空间上的连续  
2、kmalloc能分配的大小有限,vmalloc能分配的大小相对较大  
3、vmalloc比kmalloc要慢  
4、kmallloc使用的是slab[内存分配](https://www.baidu.com/s?wd=%E5%86%85%E5%AD%98%E5%88%86%E9%85%8D&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YLm1fsPWu-ujubmWnLrjcL0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3Erj04n10Ln1RYPHRdPW6vrHnz)机制，而vmalloc使用的是[伙伴系统](https://www.baidu.com/s?wd=%E4%BC%99%E4%BC%B4%E7%B3%BB%E7%BB%9F&tn=44039180_cpr&fenlei=mv6quAkxTZn0IZRqIHckPjm4nH00T1YLm1fsPWu-ujubmWnLrjcL0ZwV5Hcvrjm3rH6sPfKWUMw85HfYnjn4nH6sgvPsT6KdThsqpZwYTjCEQLGCpyw9Uz4Bmy-bIi4WUvYETgN-TLwGUv3Erj04n10Ln1RYPHRdPW6vrHnz" \t "http://zhidao.baidu.com/question/_blank)分配机制

1. platform总线设备及总线设备如何编写
2. 浅述GCC编译器在编译时都有哪几个过程

**预处理-----> 编译 ----> 汇编 ----> 链接** 无选项链接

用法：[root]# gcc hello.o –o hello.exe

作用：将编译输出文件hello.o链接成最终可执行文件hello.exe。