1. 请大致描述CPU Cache的工作原理，描述Cache Miss会带来怎样的代价，并进而给出对应的基本优化思路。
2. 请大致描述CPU流水线结构的工作特点，并给出对应的基本优化思路。
3. 关于ARM：

MOV r0, #0x3FFFC0 这句指令有错误吗？为什么？如果有错请提出修改意见

1. 请解释 ORR r0, r0, r1, LSR #16 的含义，并指出这句指令中的LSR #16会不会带来额外的开销。
2. 解释下面的指令：

<1> MLA r0, r1, r2

<2> SMULL r0, r1, r2, r3

<3> BL Label

6、把下面的指令翻译成C代码，并说出MOVLT与MOV的执行效率是否相同？

asm volatile ( "CMP r8, #0" : : );

asm volatile ( "MOVLT r8, #0" : : : "r8" );

asm volatile ( "MOVGE r8, %0" : : "r"(val) : "r8" );

1. ARM寄存器一共有哪些？其中哪些可以放心使用，哪些应该小心使用甚至不使用？
2. 常见的ARM指令集中不包括除法指令，那么对于程序中出现的除法，你认为应该如何处理？（请分为内存紧张与不紧张两种情况讨论）
3. 目前大部分手机采用的ARM的芯片，都没有浮点数协处理器，然而几乎所有的3D应用程序都需要小数类型的支持，对此有什么解决方案？
4. ARM-Linux启动分成几部分，简述流程
5. 什么是交叉编译？简述交叉编译的编译步骤和命令。
6. Linux内核的编译步骤中，make menuconfig的作用是什么，内核代码中的Kconfig和Makefile的作用是什么，他们之间的关联。
7. 嵌入式中的驱动程序编写中，经常要用到中断，有些中断程序处理的过程较长，因此我们最好不要把中断处理过程放在中断服务中，可以用什么方法来实现这种处理。
8. 简述linux驱动模型。
9. 某计算机的字长是32位，它的存储容量是1MB，按字编址，它的寻址范围是？（256K,1M,256KB,1MB）
10. CPU为什么需要中断，CPU处理终端的流程是怎么样的？

17、驱动中断

1. 主设备号的用处
2. 在指令系统的各种寻址方式中，获取操作数最快的方式是？

若操作数的地址包含在指令中，是属于那种寻址方式？

（直接寻址，立即寻址，寄存器寻址，间接寻址）

1. 将一个包含32768个基本存储单元的存储电路设计成为8位为一个字节的ROM，则ROM有多少个地址？至少需要多少根数据读出总线?
2. 在嵌入式系统的存储结构中存取速度最快的是？容量最大的是？

（Cache,RAM,Nor Flash ,Nand Flash）

22、每条指令都可以分解Wie取指、分析、执行三步骤，已知取指的时间为t = 4Q，根分析的时间t = 3Q，执行的时间为t = 5Q；

1. 如果按照串行方式执行完100条指令需要多少时间？
2. 如果按照流水线的方式执行，执行完100条指令需要多少时间？
3. 假设CPU的复位向量的地址为0X12340000，用C语言怎么实现软件的复位
4. GNU Makefile中 $@含义是什么？
5. Bash shell中$#含义是什么？
6. 给你一块网卡，写出驱动中要实现的接口函数。
7. U-boot启动的时候为什么一开机就要使用汇编，而不使用C语言？U-boot为C语言的运行要初始化哪些运行环境。
8. 重点：描述一下你写的驱动

描述一下tty设备的驱动架构

1. 内核启动时决定要挂载哪个文件系统所要取的文件是\_\_\_;
2. 为什么在内核中要使用互斥和并发机制；
3. I2C UART PCI SPI等通信协议 总线机制要非常熟析
4. 关于ARM板的关闭中断，屏蔽中断的区别，ARM启动流程分析，为什么中断要分屏蔽中断和关闭中断
5. STMFD SP！ {R0} 压栈的过程详细分析
6. LDMFD SP！ {R0-R3，R12， PC}^过程分析尤其注意^的意义。
7. ARM总线协议
8. U-boot一开始为什么要中断，而不是屏蔽中断，可以不这样做吗？
9. 写一个ARM裸机的中断处理程序的流程，具体步骤，比如键盘的敲击输入。
10. 中断里不能做什么事
11. 内核的作用
12. /poll/select/epoll的实现机制
13. 如何使移动设备系统启动加快
14. 有关于中断处理的问题，即在操作系统中中断时如何实现的
15. 怎么在Makefile中添加动态库？
16. 请问cache ,ram ,rom的区别
17. 一个15位地址线，16位双向数据线的ram的大小是多少
18. 请问为防止优先级反转，内核当中可以做什么处理。
19. linux常见文件系统
20. 设备驱动中，字符设备及其中断怎么实现
21. CPP面向对象的编程方式，怎么在C语言中实现
22. Linux中 设备以什么方式提供访问
23. 写两段代码，完成:

将a的bit3设为1

将a的bit3清零

1. 驱动里面为什么要有并发、互斥的控制？如何实现？举例说明。
2. 请简述arm linux内核启动流程。
3. 中断是嵌入式系统中重要的组成部分，这导致了很多编译开发商提供一种扩展—让标准C支持中断。具代表事实是，产生了一个新的关键字\_\_interrupt。下面的代码就使用了\_\_interrupt关键字去定义了一个中断服务子程序(ISR)，请评论一下这段代码的。
4. \_\_interrupt double compute\_area (double radius)
5. {
6. double area = PI \* radius \* radius;
7. printf("/nArea = %f", area);
8. return area;
9. }
10. CPU管理外围设备的输入输出控制方式有几种？请简述大概机理。
11. 请简述linux内核终端处理分成上半部分和下半部分的原因，为何要分？如何实现？
12. 设计并实现一个在linux平台下简单的内存FIFO字符设备驱动，并简述该驱动的验证方法。
13. 介绍串口，I2C总线，SPI各有几根线，功能，各自的特点。
14. 描述Linux的开机到登陆界面的启动过程：
15. Linux系统移植的具体流程（要具体，只说思路不算）
16. UART、I2C等模块主要特点、使用的场合。
17. 什么是异常处理机制？混合编程的原理，为什么要用到混合编程。

63、问：Linux平台下，如何让字符设备驱动支持阻塞与非阻塞IO模型，以read操作为例，编写代码简要说明

答：

ssize\_t read(struct file \*filep, char \_\_user \*ubuf, size\_t size, loff\_t \*loff)

{

        unsigned long p = \*loff;

        int count,ret;

          /\*

        调用wait\_event函数在 len!=0这个条件为假，也就是没有数据可读的时候，会将调用本read函数的进程丢到read\_queue这个等待队列里面去，让其睡眠。

当len!=0这个条件为真的时候，也就是有数据可读的时候，该函数(wait\_event)会立即返回，不会将

进程放到等待队列里面，也就是说不让该进程睡眠，形成不了阻塞。

          \*/

        wait\_event\_interruptible(read\_queue, (len!=0));

        count = size > (GLOBALMEM\_SIZE-p) ? (GLOBALMEM\_SIZE - p):size;

        count = count > len ? len : count;

    copy\_to\_user(ubuf, mem+p , count);

}