Linux内核启动：BIOS启动阶段Linux认证考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/644/2021_2022_Linux_E5_86_85_E6_A0_c103_644665.htm Linux 内核代码复杂、庞大，让人感觉难以入手，正是因为它的复杂性，任何一本教材都会把相关的内容进行分类讲解，例如中断处理，文件系统，等等。然而在阅读相关章节时，你是不是常常想弄明白某个相关的数据结构是在什么时候建立的?是在什么时候初始化的?本章从BIOS启动开始，对内核的启动部分进行逐步介绍，这样可以为读者建立一个初步的、整体的认识。读者在进一步的学习过程中，可以根据本章的介绍迅速找到相关内容的初始化部分。 ◆BIOS 启动阶段 CPU在上电初始化时，指令寄存器CS:EIP总是被初始化为固定的值，这就是CPU复位后的第一条指令的地址。断电后内存中的内容就丢失了，所以这一条指令必须保存在“ 非易失 ”的存储器中。此类存储器包括ROM，PROM，EPROM，Nor Flash等。早期的BIOS存放在只读存储器中，非常不方便修改。现在EPROM和Nor Flash都能够通过电的方式来进行擦除和编程写入，所以通常升级BIOS就是利用BIOS芯片的电可擦除编程特性。对于32位地址总线的系统来说，4GB的物理地址空间至少被划分为两个部分，一部分是内存的地址空间，另外一部分地址空间用于对BIOS芯片存储单元进行寻址。除此之外，随着系统外部设备的增加以及设备本身的板载存储空间的增加，16位8086处理器拥有的64KB的IO地址空间早已不够(通过in/out汇编指令来访问的I/O端口。)，实际上4GB的物理内存地址空间还有一部分用于外部设备的板载存储空间的寻址。x86复位后工作在

实模式下，该模式下CPU的寻址空间为1MB。 CS:IP的复位值是FFFF:0000，物理为FFFF0。主板的设计者必须保证把这个物理地址映射到BIOS芯片上，而不是RAM上。 早期的IBM PC地址空间映射如图4.1所示。其中高256KB的只读存储空间映射到BIOS芯片中，中间的128KB VVDR映射到视频卡的存储空间，屏幕上面的像素点受该区域控制，剩下的640KB映射到RAM上面。可以看出对于硬件系统的设计者来说，物理地址空间也是一种资源，而这里所说的映射就是以硬件方式对物理地址资源的分配。图4.1所示的640KB的RAM是BIOS设计者自由使用的区域，如何使用取决于 BIOS软件的设计者 。CPU执行BIOS代码对系统进行必要的初始化，并在物理地址0开始的1KB内存中建立实模式下的中断向量表，随后的一部分内存被用来保存BIOS在启动阶段检测到的硬件信息。另外BIOS代码在执行期还需要使用随后的一部分内存。最后BIOS会根据配置把引导设备的第一个扇区加载到物理地址0x07C00的地方，然后跳转到这里继续执行。通常这是Boot Loader的代码，Boot Loader接着把内核加载到内存中。前面说过arch/x86/boot/tools/build工具把setup和vmlinux合成一个bzImage。setup是实模式的代码，vmlinux是保护模的代码。BIOS把Boot Loader加载到0x07C00的地方然后跳转到这里继续执行，之后Boot Loader会把实模式代码setup加载到0x07C00之上的某个地址上，其中setup的前512字节是一个引导扇区，现在这个引导扇区的作用并不是用来引导系统，而是为了兼容及传递一些参数。之后Boot Loader会跳转到setup的入口点，通过前面对链接脚本arch/x86/boot/setup.ld的分析(见第3.3节)，我们知道这个入口点是_start。linux认证网，加入收藏