你管这破玩意叫操作系统源码 | 第一回 最开始的两行代码

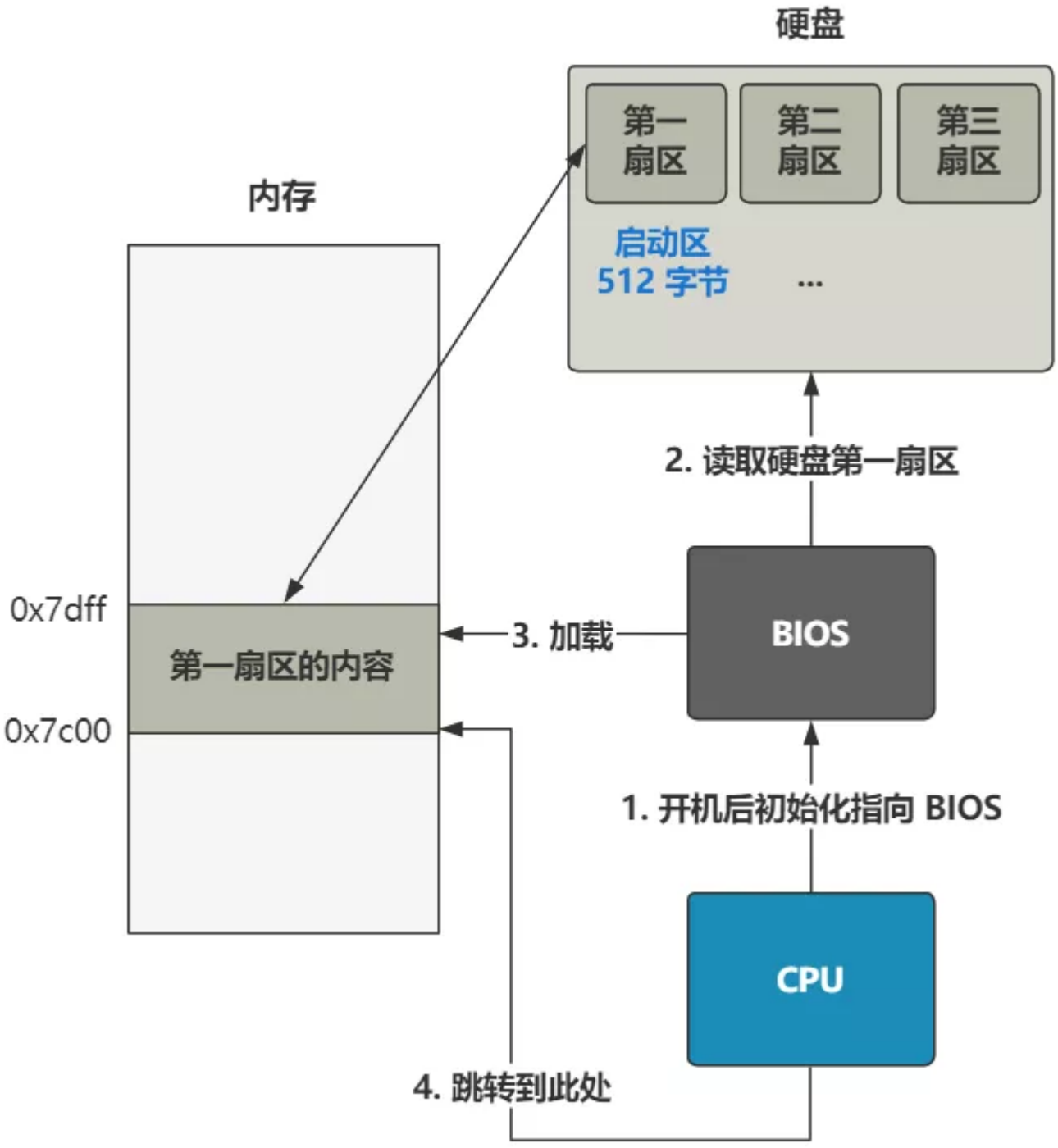
Original 闪客sun 低并发编程 2021-11-11 08:30

收录于话题
#你管这破玩意叫操作系统源码
20个 >

从这一篇开始，您就将跟着我一起进入这操作系统的梦幻之旅！

别担心，每一章的内容会非常的少，而且你也不要抱着很大的负担去学习，只需要像读小说一样，跟着我一章一章读下去就好。

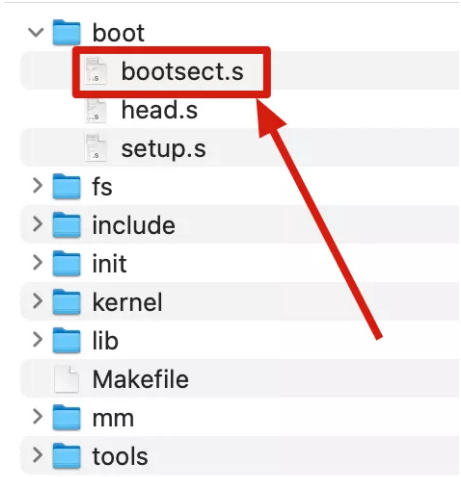
话不多说，直奔主题。当你按下开机键的那一刻，在主板上提前写死的固件程序 **BIOS** 会将硬盘中**启动区的 512 字节**的数据，原封不动复制到**内存中的 0x7c00** 这个位置，并跳转到那个位置进行执行。


硬盘
第一扇区
第二扇区
第三扇区
启动区
512 字节
...
内存
0x7dff
0x7c00
第一扇区的内容
2. 读取硬盘第一扇区
3. 加载
BIOS
1. 开机后初始化指向 BIOS
CPU
4. 跳转到此处

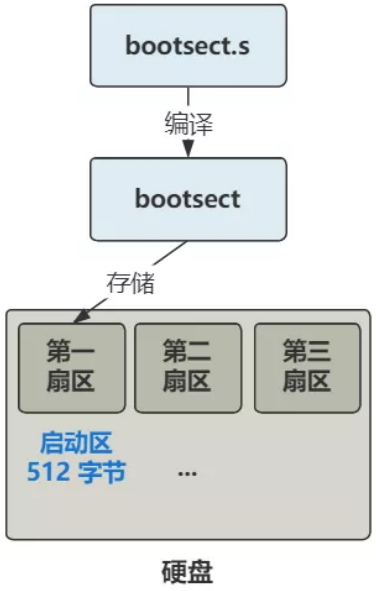
启动区的定义非常简单，只要硬盘中的 0 盘 0 道 1 扇区的 512 个字节的最后两个字节分别是 **0x55** 和 **0xaa**，那么 BIOS 就会认为它是个启动区。

所以对于我们理解操作系统而言，此时的 BIOS 仅仅就是个代码搬运工，把 512 字节的二进制数据从硬盘搬运到了内存中而已。**所以作为操作系统的开发人员，仅仅需要把操作系统最开始的那段代码，编译并存储在硬盘的 0 盘 0 道 1 扇区即可。**之后 BIOS 会帮我们把它放到内存里，并且跳过去执行。

而 Linux-0.11 的最开始的代码，就是这个用汇编语言写的 **bootsect.s**，位于 **boot** 文件夹下。



通过编译，这个 bootsect.s 会被编译成二进制文件，存放在启动区的第一扇区。



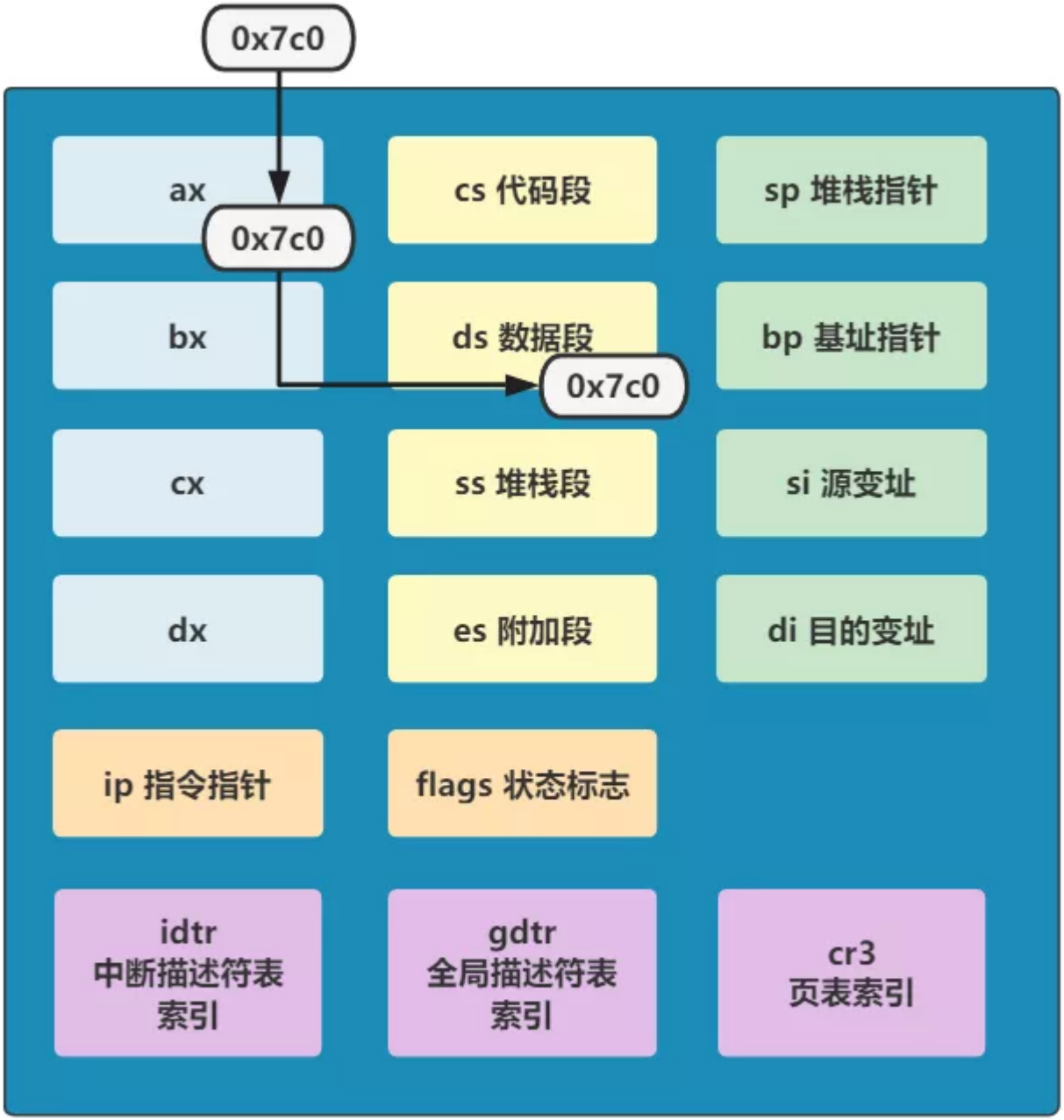
随后就会如刚刚所说，由 BIOS 搬运到内存的 0x7c00 这个位置，而 CPU 也会从这个位置开始，不断往后一条一条语句无脑地执行下去。

那我们的梦幻之旅，就从这个文件的第一行代码开始啦！

```
mov ax,0x07c0
mov ds,ax
```

好吧，先连续看两行。

这段代码是用汇编语言写的，含义是把 0x07c0 这个值复制到 **ax 寄存器**里，再将 ax 寄存器里的值复制到 **ds 寄存器**里。那其实这一番折腾的结果就是，让 ds 这个寄存器里的值变成了 0x07c0。


0x7c0
ax
0x7c0
cs 代码段
sp 堆栈指针
bx
ds 数据段
bp 基址指针
0x7c0
cx
ss 堆栈段
si 源变址
dx
es 附加段
di 目的变址
ip 指令指针
flags 状态标志
idtr
中断描述符表
索引
gdtr
全局描述符表
索引
cr3
页表索引

CPU 中的关键寄存器

ds 是一个 16 位的段寄存器，具体表示数据段寄存器，在内存寻址时充当段基址的作用。啥意思呢？就是当我们之后用汇编语言写一个内存地址时，实际上仅仅是写了偏移地址，比如：

```
mov ax, [0x0001]
```

实际上相当于

```
mov ax, [ds:0x0001]
```

ds 是默认加上的，表示在 ds 这个段基址处，往后再偏移 0x0001 单位，将这个位置的内存数据，复制到 ax 寄存器中。

形象地比喻一下就是，你和朋友商量去哪玩比较好，你说天安门、南锣鼓巷、颐和园等等，实际上都是**偏移地址**，省略了北京市这个**基址**。

当然你完全可以说北京天安门、北京南锣鼓巷这样，每次都加上北京这个前缀。不过如果你事先和朋友说好，以下我说的地方都是北京市里的哈，之后你就不用每次都带着北京市这个词了，是不是很方便？

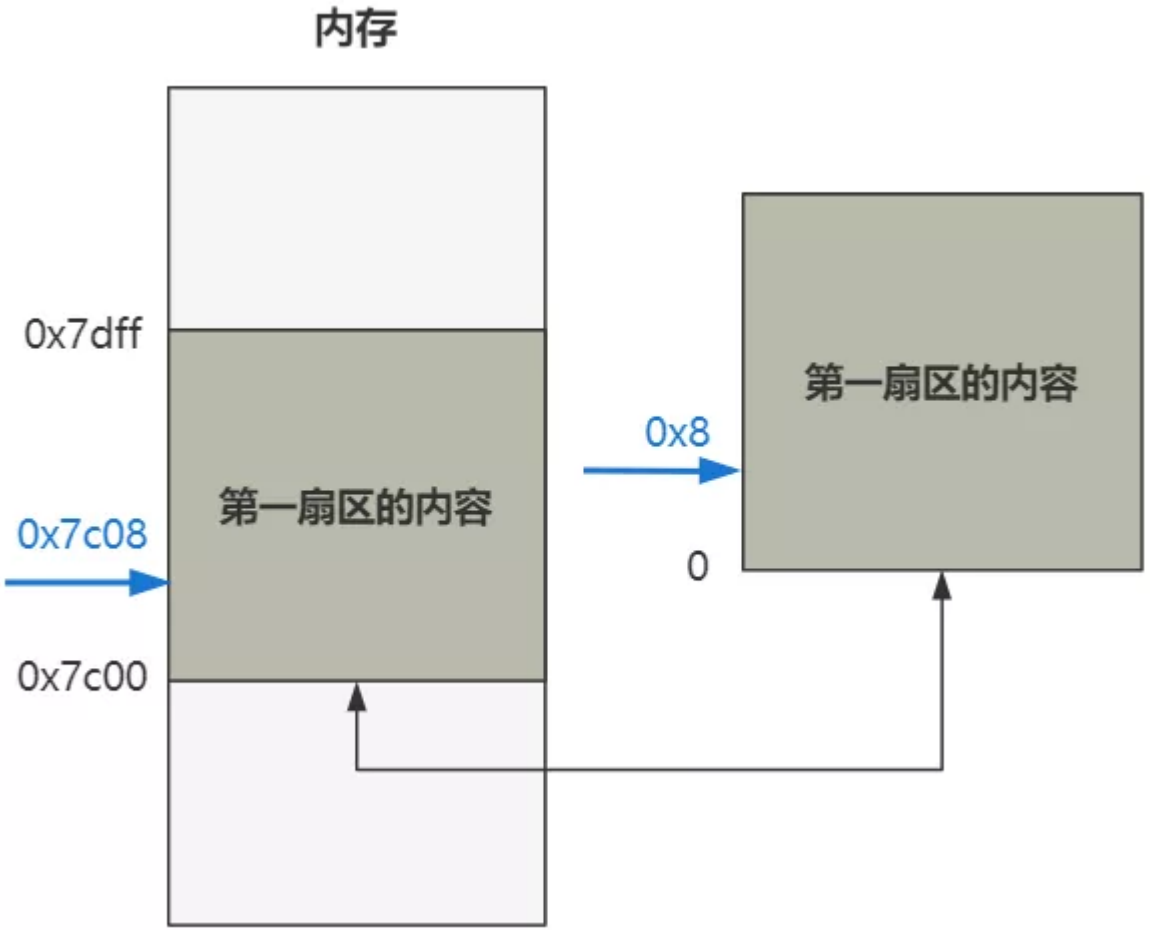
那 ds 这个数据段寄存器的作用就是如此，方便了描述一个内存地址时，可以省略一个基址，没什么神奇之处。

ds：0x0001
北京市：南锣鼓巷

再看，这个 ds 被赋值为了 0x07c0，由于 x86 为了让自己在 16 位这个实模式下能访问到 20 位的地址线这个历史因素（不了解这个的就先别纠结为啥了），所以段基址要先左移四位。**那 0x07c0 左移四位就是 0x7c00**，那这就刚好和这段代码被 BIOS 加载到的内存地址 0x7c00 一样了。

也就是说，之后再写的代码，里面访问的数据的内存地址，都先默认加上 0x7c00，再去内存中寻址。

为啥统一加上 0x7c00 这个数呢？这很好解释，BIOS 规定死了把操作系统代码加载到内存 0x7c00，那么里面的各种数据自然就全都被偏移了这么多，所以把数据段寄存器 ds 设置为这个值，方便了以后通过这种基址的方式访问内存里的数据。



OK，赶紧消化掉前面的知识，那本篇就到此为止，只讲了两行代码，知识量很少，我没骗你吧。

希望你能做到，对 BIOS 将操作系统代码加载到内存 0x7c00，以及我们通过 mov 指令将默认的数据段寄存器 ds 寄存器的值改为 0x07c0 方便以后的基址寻址方式，这两件事在心里认可，并且没有疑惑，这才方便后面继续进行。

后面的世界越来越精彩，欲知后事如何，且听下回分解。

------- 本回扩展资料 -------

有关寄存器的详细信息，可以参考 Intel 手册：
Volume 1 Chapter 3.2 OVERVIEW OF THE BASIC EXECUTION ENVIRONMEN

有关计算机启动部分的原理如果还不清楚，可以看我之前的一篇文章了解一下：
计算机的启动过程

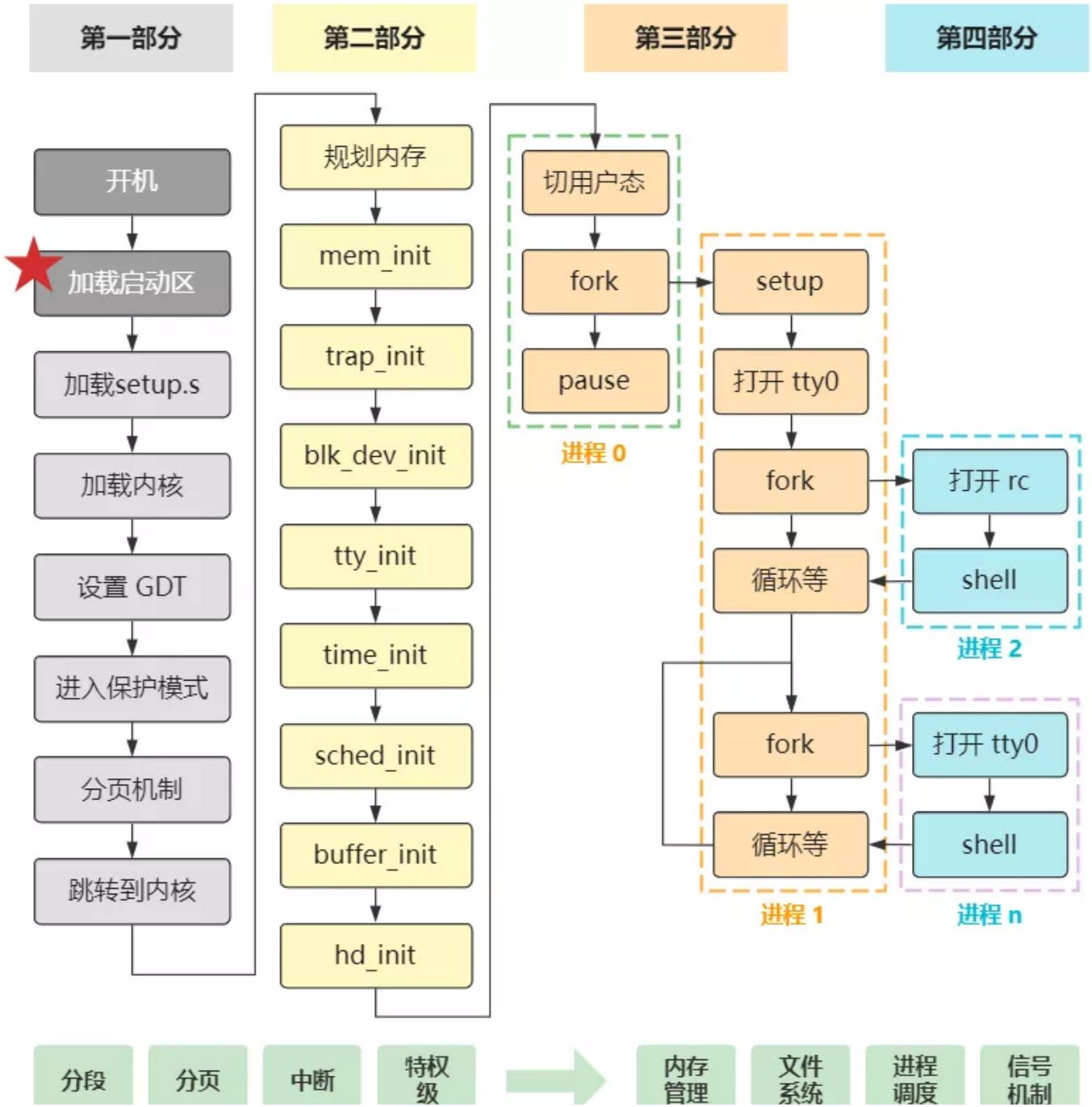
如果想了解计算机启动时详细的初始化过程，还是得参考 Intel 手册：
Volume 3A Chapter 9 PROCESSOR MANAGEMENT AND INITIALIZATION

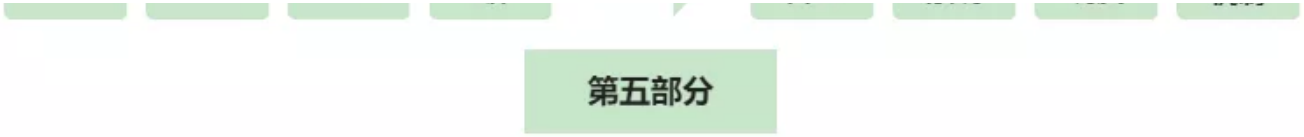
------- 关于本系列 -------

本系列的开篇词看这
闪客新系列！你管这破玩意叫操作系统源码

本系列的扩展资料看这（也可点击**阅读原文**）
https://github.com/sunym1993/flash-linux0.11-talk

本回的内容在全局视角中的星星那里





最后，祝大家都能追更到系列结束，只要你敢持续追更，并且把每一回的内容搞懂，我就敢让你在系列结束后说一句，我对 Linux 0.11 很熟悉。

另外，本系列完全免费，希望大家能多多传播给同样喜欢的人！我们下回见。

收录于话题 #你管这破玩意叫操作系统源码 20

‹ 上一篇

闪客新系列！你管这破玩意叫操作系统源码

下一篇 ›

你管这破玩意叫操作系统源码 | 第二回 自己给自己挪个地儿

Read more

People who liked this content also liked

第17回 | 原来操作系统获取时间的方式也这么 low

低并发编程