



## linux 3.6 启动源码分析(四) rest\_init

原创

2013年12月16日 14:25:13

1310

在内核初始化函数start\_kernel执行到最后，就是调用rest\_init函数，这个函数的主要使命就是创建并启动内核线程init。这个函数虽然意思为剩下的初始化，但是这个“剩下”的可是内容颇多，下面详细分析如下：

```
[cpp]
1. static ninline void __init_refok rest_init(void)
2. {
3.     int pid;
4.     rcu_scheduler_starting();// 1.内核RCU锁机制调度启动,因为下面就要用到
5.     /*
6.      * We need to spawn init first so that it obtains pid 1, however
7.      * the init task will end up wanting to create kthreads, which, if
8.      * we schedule it before we create kthreadd, will OOPS.
9.      * 我们必须先创建init内核线程，这样它就可以获得pid为1。
10.     尽管如此init线程将会挂起来等待创建kthreads线程。
11.     如果我们在创建kthreadd线程前调度它，就将会出现OOPS。
12.     */
13.     kernel_thread(kernel_init, NULL, CLONE_FS | CLONE_SIGHAND);
14.     numa_default_policy();// 1.设定NUMA系统的内存访问策略为默认
15.     pid = kernel_thread(kthreadd, NULL, CLONE_FS | CLONE_FILES);
16.     /*
17.     1.创建kthreadd内核线程，它的作用是管理和调度其它内核线程。
18.     2.它循环运行一个叫做kthreadd的函数，该函数的作用是运行kthread_create_list全局链表中维护的内核线程。
19.     3.调用kthread_create创建一个kthread，它会被加入到kthread_create_list 链表中；
20.     4.被执行过的kthread会从kthread_create_list链表中删除；
21.     5.且kthreadd会不断调用scheduler函数让出CPU。此线程不可关闭。
22.     */
23.     上面两个线程就是我们平时在Linux系统中用ps命令看到：
24.     $ ps -A
25.     PID TTY TIME CMD
26.     3.1 ? 00:00:00 init
27.     4.2 ? 00:00:00 kthreadd
28.     */
29.     rcu_read_lock();
30.     kthreadd_task = find_task_by_pid_ns(pid, &init_pid_ns);
31.     rcu_read_unlock();
32.     complete(&kthreadd_done);
33.
34.     /*1.获取kthreadd的线程信息，获取完成说明kthreadd已经创建成功。并通过一个
35.     complete变量（kthreadd_done）来通知kernel_init线程。*/
36.     /*
37.     * The boot idle thread must execute schedule()
38.     * at least once to get things moving:
39.     */
40.     init_idle_bootup_task(current);
41.     schedule_preempt_disabled();
42.     /* Call into cpu_idle with preempt disabled */
43.     cpu_idle();
44. }
```

在以上的函数中，内核创建了两个内核线程，一个是内核线程的管理者，另一个是内核初始化线程init，后者是我们分析内核启动需要关注的，这个线程继续做系统的初始化（其中就包含了设备驱动系统）



严禁讨论涉及中国之军/政相关话题，违者会被禁言、封号！

### linux 3.6 启动源码分析(一)



qing\_ping

2013年12月16日 13:02

2080

作为需要和硬件打交道的工程师来说，比较关注的是驱动和CPU初始化这一块。所以我沿着启动的路线，重点学习一下和硬件相关的代码。就从linux解压的入口说起。学习阶段，基本是参考大神文章http://bl...

### linux启动流程（从start\_kernel中的rest\_init函数到init进程（1））

linux启动流程（从start\_kernel中的rest\_init函数到init进程（1）） 在init/main.c文件中有个函数叫start\_kernel，它是用来启动内核的主函数，我...



zhongyh

2013年04月23日 23:35

1892

### 程序员不会英语怎么办？

老司机教你一个数学公式秒懂天下英语



### CentOs7 rest\_init 0x80 解决方案



n664500560

2017年09月19日 21:37

325

Two ways to fix the issue with kernel-3.10.0-327\* : - for installed system : - boot with the ini...

### linux 3.6 启动源码分析(二) start\_kernel



qing\_ping

2013年12月16日 13:38

2910

在构架相关的汇编代码运行完之后，程序跳入了构架无关的内核C语言代码：init/main.c中的start\_kernel函数，在这个函数中Linux内核开始真正进入初始化阶段，进行一系列与内核相关的初...



qing\_ping

关注

原创

粉丝

喜欢

评论

30

31

1

1

等级：博客

访问量：3万+

积分：650

排名：7万+



#### 他的最新文章

[更多文章](#)
[Linux设备模型（四）class](#)
[Linux设备模型（三）platform](#)
[Linux设备模型（二）上层容器](#)
[linux 设备模型\(一\)对象层](#)
[Linux中断子系统-中断接口](#)

#### 文章分类

[linux开发](#)

2篇

[linux 驱动学习](#)

3篇

[linux源码学习](#)

14篇

#### 文章存档

[2014年1月](#)

2篇

[2013年12月](#)

12篇

[2013年11月](#)

1篇

[2013年10月](#)

4篇

[2013年9月](#)

9篇

[2012年2月](#)

1篇

[展开](#)

#### 他的热门文章

[linux 3.6 启动源码分析\(五\) kernel\\_init进程](#)

3057

[linux 3.6 启动源码分析\(二\) start\\_kernel](#)

2909

[linux 3.6 启动源码分析\(七\) do\\_initcalls](#)

2454

[linux 3.6 启动源码分析\(一\)](#)

2078

[linux 3.6 启动源码分析\(三\) setup\\_arch](#)

1893

[嵌入式linux 运行期间升级u-boot, kernel和文件系统](#)

1792

[linux 3.6 启动源码分析\(六\) do\\_basic\\_setup](#)

1681

[linux下读写u-boot环境变量](#)

1354

[Linux中断子系统-中断初始化](#)

1329

[linux 3.6 启动源码分析\(四\) rest\\_init](#)

1307